Machine learning approaches to rank news feed updates on social media

Sami Belkacem

People's Democratic Republic of Algeria
Ministry of Higher Education and Scientific Research
University of Science and Technology HOUARI BOUMEDIENE
Faculty of Electronics and Computer Science

PhD Thesis

Presented to obtain the Doctorate degree

In : Computer Science

Specialty : Artificial Intelligence

By : BELKACEM Sami

Subject

Machine learning approaches to rank news feed updates on social media

Publicly defended, on 12/04/2021, before a jury composed of :

M.	AZZOUNE Hamid	Professor	at USTHB	President
M.	BOUKHALFA Kamel	Professor	at USTHB	Thesis supervisor
M.	BOUSSAID Omar	Professor	at Univ of LYON 2	Thesis co-supervisor
M.	BOUYAKOUB Fayçal	Professor	at USTHB	Examiner
M.	VALENTE de OLIVEIRA José	HDR	at Univ of Algarve	Examiner
Mrs.	ALIANE Hassina	MRA	at CERIST	Guest

To my parents. ...

Acknowledgements

First of all, I thank the almighty god Allah, who gave me the ability, the strength, and the patience to accomplish this modest work.

I would like to express my deep gratitude and sincere thanks to my supervisors, Prof. Boukhalfa Kamel and Prof. Boussaid Omar, for the scientific contribution, the time and effort they devoted to my supervision, the invaluable support, and the appropriate assistance they so kindly provided to conduct this work during all years of the thesis project.

I would like to thank and express my sincere appreciation for the jury members, namely the jury president, Prof. Azzoune Hamid, and the examiners, Prof. Bouyakoub Fayçal, Prof. Valente de Oliveira José, and Dr. Aliane Hassina, for the time and effort they devoted to reading my thesis and the honor they do to me by agreeing to evaluate my thesis dissertation. I am very glad and honored to be examined by such high-level experts.

I would like to thank each of my former teachers at USTHB university for the noble work they accomplish every day, transferring their knowledge and expertise to students. I had great teachers and mentors during my years at USTHB, each one of them contributed to the knowledge I have acquired so far.

I would like to express my gratitude to all my friends and colleagues at USTHB university, especially Hadj Ameur Mohamed, Sellam Abdallah, and Brahimi Mohamed, for their moral support, precious advice, and help during the whole period of this project.

I cannot end without thanking my family, my sister, brother, parents, and all members of the small and large family. Thank you all for always being by my side.

To all of you, thank you.

Abstract

Today, social media such as *Facebook* and *Twitter* are used by hundreds of millions of users worldwide. Due to the large number of members and the large amount of data posted and shared (messages, articles, videos, music, photos, etc.), users find themselves overwhelmed by a large volume of updates in their news feed, also known as *the news stream*. The news feed is typically displayed in reverse chronological order from the most recent to the least recent update. In addition, several research works have shown that the majority of the updates are considered irrelevant. Therefore, large data volume and irrelevance make it difficult for users to quickly catch up and interact with updates that may be of interest to them. To this end, based on the prediction of a relevance score between a user and a new update unread in the news feed, research work has proposed approaches to rank and display news feed updates in descending relevance order. These approaches aim to provide recommendations and help users quickly find relevant updates. In this thesis work, we first carry out a state-of-the-art of the proposed approaches in the field of ranking news feed updates according to several criteria: the features that may influence the relevance of updates, the relevance prediction models, the training and evaluation of prediction models, the target social media platforms, etc. The goal is first to show the advantages of the proposed approaches, their limitations, and identify open research issues. Thereafter, we propose and implement several intelligent machine-learning-based models to address the limitations of the approaches proposed in the literature. The objective of the proposed approaches is to collect and preprocess real data, extract the features that may influence relevance, train personalized prediction models for each user, predict relevance scores of news feed updates, and finally conduct in-depth experiments to evaluate and validate the proposed models.

Résumé

De nos jours, les réseaux sociaux tels que *Facebook* et *Twitter* sont utilisés par des centaines de millions d'utilisateurs à travers le monde. En raison du nombre important de membres et de la grande quantité d'informations postées et partagées (messages, articles, vidéos, musique, photos, etc.), les utilisateurs se retrouvent submergés par un grand volume d'informations dans leur fil d'actualité, appelé également *flux de nouvelles*. Le fil d'actualité est généralement affiché par ordre chronologique inverse, de l'information la plus récente à la moins récente. De plus, plusieurs travaux de recherche ont montré que la majorité de ces informations sont considérées comme non pertinentes. De ce fait, le grand volume de données et leur non-pertinence rendent difficile pour les utilisateurs de consulter et d'interagir rapidement avec les informations susceptibles de les intéresser. À cet effet, en se basant sur la prédiction d'un score de pertinence entre un utilisateur et une nouvelle information non consultée dans son fil d'actualité, des travaux de recherche ont proposé des approches pour trier et afficher les informations des fils d'actualité par ordre décroissant de pertinence. L'objectif de ces approches étant de fournir des recommandations et aider les utilisateurs à consulter rapidement les actualités pertinentes. Dans ce travail de thèse, nous effectuons d'abord un état de l'art des approches proposées dans le domaine du tri des fils d'actualité selon plusieurs axes: les facteurs influençant la pertinence des informations, les modèles de prédiction de la pertinence, l'apprentissage et l'évaluation des modèles de prédiction, les réseaux sociaux cibles, etc. Le but dans un premier temps étant de montrer les avantages des approches proposées, leurs limites, et identifier des ouvertures de recherche. Par la suite, nous proposons et implémentons plusieurs modèles intelligents basés sur des techniques d'apprentissage automatiques pour répondre aux limites des approches proposées dans la littérature. L'objectif des approches proposées est de collecter et prétraiter des données réelles, d'extraire les facteurs susceptibles d'influencer la pertinence, d'entrainer des modèles personnalisés de prédiction pour chaque utilisateur, de prédire les scores de pertinence des nouvelles actualités, et enfin d'effectuer des expérimentations approfondies pour évaluer et valider les modèles proposés.

Table of contents

List of figures

List of tables

Chapter 1

Introduction

1.1 Motivation

In the new century, the possibility for everyone to easily publish and share content with the public through the World Wide Web has changed communication. For instance, social media has transformed mass communication from the monopoly of the unidirectional and traditional mass media such as the printing press, television, and radio to multidirectional communication where everyone can participate. Indeed, social media facilitated a shift from a consumer-oriented communication culture to a communication culture of participation, from a world in which a small number of people report, create, decide, and form opinions to a digital world in which everyone has opportunities to participate. Nowadays, user-generated content made publicly available online becomes an alternative news source. In combination with the achievements of the last decade in hardware development, equipping a large part of the world population with mobile devices with cameras and mobile access to the Internet, anyone can post something as it happens to make it news if only enough users consider it relevant.

Nowadays, social media such as *Facebook* and *Twitter* produce massive streams that can be analyzed for a wide variety of insights [1]. Indeed, such streams are referred to as *news feeds*, which represent the primary feature through which users are exposed to the latest updates posted on social media platforms. The news feed is typically displayed in reverse chronological order from the most recent to the least recent update and is particularly rich in terms of the large number of updates that users can produce

over time [2]. For example, according to the official social media report of 2020[1], the average number of tweets sent per day is 500 million, which is around 5790 tweets per second. Analogously, there are about 2.7 billion users on *Facebook*, with 67% being daily active users. The analysis of massive data in news feeds poses numerous challenges that are unique in social media analysis and mining [3]:

- The large amount of data that is continuously produced over time in large-scale applications such as social media results in massive data streams.
- The social media data can be irrelevant and unreliable such that methods are required to analyze the relevance and reliability of the data.
- The data often contains social structural information in addition to the textual content of data such as social relationships and social interactions.
- The user-generated content on social media is short and noisy, unlike traditional documents. For example, a tweet on *Twitter* is limited to 280 characters.
- The patterns in the data may evolve significantly over time. Therefore, the analysis must be done dynamically in real-time.

The rapidly growing volume of data available on social media and their unique specificity make it increasingly difficult for users to find the content they are looking for in the news feed [4]. Indeed, from the tremendous amount of news feed updates available online, users need to browse a large number of updates and then evaluate their relevance. Furthermore, social media users are not only interested in updates that refer to a specific topic. Independently of the topic, users may be interested in criteria such as the popularity of the update and the social ties that link them to the person who posted the update [5]. However, traditional search engines based on query-dependent ranking do not sufficiently support users in this task because they have two major weaknesses [1]: they require the user to specify search terms and neglect the specificity of user-generated content on social media. Therefore, the challenge is to develop solutions that help users cope with the massive amount of data in news feeds. For instance, predict the relevance of news feed updates and establish a ranking based on how likely users will find given updates interesting.

In recent years, the news feed has been studied extensively, not just in the text

[1] https://wearesocial.com/digital-2020

domain but also in the context of a wide variety of multi-dimensional applications such as numerical and categorical data [6]. For example, the problem of performing recommendations in social news feeds on *Twitter* has been studied in [7]. Indeed, because of the sheer volume of such news feeds, it is crucial to present a small number of target recommendations to recipient users. Moreover, in several research approaches, ranking news feed updates in descending relevance order has been proposed as a solution to help users quickly catch up with the relevant updates [8]. Unlike the chronological news feed, the ranking process is done such that the most relevant updates are found at the top of the news feed and the least relevant at the bottom [9]. To predict the relevance, a large number of studies have investigated the combination of social, content, and temporal features in different models [10]. These prediction models have to define and extract the most suitable and efficient features. However, the field of ranking news feed updates is still in its infancy, and significant scope exists in improving and designing new methods that address the limitation of the approaches proposed in the literature.

In the last decade, recommender systems and online social networks have established strong cooperation. Indeed, social media and recommendation systems both aim at coping with the huge amount of data produced and shared by users through online platforms, trying to maintain a high user engagement [11]. Furthermore, over the last years, artificial intelligence and machine learning techniques have gained increasing attention to handle the ever-growing data generated by various social media applications [12]. The advancement of the machine learning field itself relies on large datasets, and social media is an ideal data source for developing and testing new machine learning techniques for the academic and industry communities [13]. The aim of this thesis work is to propose machine-learning-based approaches to rank, recommend, and display news feed updates in descending relevance order based on the prediction of a relevance score between a user and a new update unread in the news feed.

1.2 Objectives

The overall objectives of this thesis are as follows:

- First, carry out a state-of-the-art of the classical approaches in recommender systems in general, and the proposed approaches in the field of ranking news feed updates in particular.

- Second, compare and analyze the approaches proposed in related work according to several criteria to show their advantages, their limitations, and identify open research issues to which we will make contributions.
- Third, propose and implement several machine-learning-based models to predict the relevance of news feed updates and address the limitations of the approaches proposed in the literature.
- After that, collect and preprocess data in a programmed way on a social media platform such as *Facebook* or *Twitter* to evaluate and validate the proposed approaches with real data.
- Then, extract the features that may influence relevance, train personalized prediction models for each user, and predict the relevance scores of news feed updates.
- Finally, conduct in-depth tests and experiments to evaluate and validate the proposed models.

1.3 Key contributions

The main contributions of this thesis are as follows:

- A survey on ranking news feed updates that was crowned by two international conference papers, namely CSA 2016 and EDA 2017.
- Prospects for proposing several machine-learning-based models to address the limitations of the approaches proposed in the literature.
- Data collection and preprocessing on *Twitter* to evaluate and validate the proposed approaches with real data.
- Implementation and evaluation of several machine-learning-based approaches to predict the relevance and rank news feed updates.
- An approach that leverages the user's expertise to predict the relevance was published in the international conference EDA 2018.
- Extension of the conference paper EDA 2017 in the Web of Science journal *International Journal of Data Warehousing and Mining (IJDWM)*.

- Extension of the conference paper EDA 2018 in the Web of Science journal *Custer Computing*.

- A comparison of seven supervised prediction models was published in the international conference EGC 2020.

- A comparison of personalized and non-personalized prediction models will be soon published.

- Prospects for further improving the proposed approaches in particular, and the field of ranking news feed updates in general.

1.4 Organization of the Thesis

The organization of this thesis is as follows:

- Chapter 2 first presents and defines social media, including a brief history and the main challenges due to the increasing amount of data. Then, the chapter provides background on recommender systems and describes three main classical recommendation approaches: content-based filtering, collaborative filtering, and hybrid filtering. Finally, the chapter introduces and presents the machine learning aspect, which is becoming increasingly important in the recommendation. In the machine learning aspect, the chapter discusses the two main approaches used in the learning process: supervised learning such as classification and regression, and unsupervised learning such as clustering and dimensionality reduction.

- Chapter 3 first provides background on ranking news feed updates on social media, including defining news feeds, presenting statistics on data volume and irrelevance that confirm the need for ranking, and formalizing the ranking process. Then, the chapter discusses work carried out in the field of ranking news feed updates in both industrial and academic communities. Finally, the chapter analyzes and compares the research works and exposes their advantages and limitations according to four main criteria: (1) the features that may influence relevance; (2) the relevance prediction models; (3) the training and evaluation methods; and (4) the evaluation platforms. Finally, the chapter identifies several open research issues to which we make contributions.

- Chapter 4 first provides a reminder of the classical non-personalized approach used in related work to rank news feed updates, which uses a prediction model for all users. Then, to predict the relevance of news feed updates and improve user experience, we use the *random forest* algorithm to train and introduce a personalized prediction model for each user. Finally, we conduct a comparative study by evaluating, analyzing, and comparing personalized and non-personalized models according to six criteria: (1) the overall prediction performance of both approaches to get a global overview of the most effective model; (2) the amount of data in the training set to investigate the robustness of each model; (3) the cold-start problem, which is a common problem in recommender systems; (4) the incorporation of user preferences over time; (5) the model fine-tuning to investigate the manageability of each model; and (6) the personalization of feature importance for users.

- Chapter 5 first describes the context of the comparison on *Twitter* according to a personalized approach that predicts the relevance of news feed updates. Then, we select and describe seven supervised algorithms that have been used in related work to predict the relevance. After that, we define a rigorous and fair comparison methodology by selecting the best parameters of each algorithm. Finally, to determine the most suitable models, we conduct a comparative study by evaluating, analyzing, and comparing the selected algorithms according to three criteria: (1) overall prediction performance of all algorithms to get a global overview of the most effective models; (2) prediction performance on various training set sizes to investigate scalability and the impact of data size on model performance; and (3) the computing speed performance to have an insight into the fastest models for eventual deployment in production.

- Chapter 6 first analyzes and discusses research work in the field of expert finding on social media. Then, we propose a personalized approach that predicts the relevance of news feed updates using supervised prediction models based on *random forest*. The proposed approach leverages the author's expertise that we infer from the biography and the textual content the author has posted, in addition to the four types of features used in related work: (1) the relevance of the update content to the recipient's interests; (2) the social tie strength between the recipient and the author; (3) the author's authority; and (4) the update

quality. Finally, we conduct experiments to evaluate the proposed approach and investigate the contribution of expertise to rank news feed updates.

- The conclusion (7) summarizes the main contributions, highlights the key results, and discusses the advantages and limitations of the work. It also provides some suggested directions and improvements for future research in the field of ranking news feed updates.

1.5 List of publications

The work carried out in this thesis has been published in the following papers:

Journal papers

1. Sami Belkacem, Kamel Boukhalfa and Omar Boussaid, Expertise-aware news feed updates recommendation: a random forest approach, Journal of Cluster Computing, 23(3), 2375-2388, November 2019, DOI: 10.1007/s10586-019-03009-w
   Publisher: Springer
   Indexed in: Web of Science
   Impact factor: 3.458

2. Sami Belkacem and Kamel Boukhalfa, Ranking News Feed Updates on Social Media: A Review and Expertise-Aware Approach, International Journal of Data Warehousing and Mining (IJDWM), Volume 17, Issue 1, January 2021, DOI: 10.4018/IJDWM.2021010102
   Publisher: IGI Global
   Indexed in: Web of Science
   Impact factor: 0.727

Conference papers

1. Sami Belkacem, Kamel Boukhalfa, and Omar Boussaid. News feeds triage on social networks: A survey. In Proceedings of the 2nd International Conference on Computing Systems and Applications (CSA), pages 34–43, December 13-14, 2016, Algiers, Algeria.

2. Sami Belkacem, Kamel Boukhalfa, and Omar Boussaid, Tri des actualités sociales: État de l'art et Pistes de recherche, EDA - Conference on Business Intelligence & Big Data, May 03-05, 2017, Lyon, France. vol. RNTI-B-13, pp.85-100.

3. Sami Belkacem, Kamel Boukhalfa, and Omar Boussaid, Leveraging expertise in news feeds: A Twitter case study, EDA - Conference on Business Intelligence & Big Data, October 04-06, 2018, Tanger, Morocco. vol. RNTI-B-14, pp.1-16.

4. Sami Belkacem, Omar Boussaid, and Kamel Boukhalfa, Ranking news feed updates on social media: A comparative study of supervised models, EGC - Conference on Knowledge Extraction and Management, January 27-31, 2020, Brussels, Belgium, vol. RNTI-E-36, pp.499-506.

Part I

State-of-the-art

Chapter 2

Social media and recommender systems

2.1 Introduction

Nowadays, every aspect of human life has been widely affected by social media. Every day, an enormous amount of data and user-generated content are uploaded to online social networks, which are analyzed and employed to improve the services provided by these platforms [14]. Recommender systems and online social networks have established strong cooperation in the last few years [15]. Indeed, both social media and recommendation systems aim to cope with the huge amount of data produced and shared by users through online platforms, trying to maintain a high user engagement [11]. This cooperation is built upon the advantages that both systems can achieve: the increasing request for personalization and the optimization of recommendation techniques by exploiting the characterization of users and content derived from social media [16]. Recommender systems targeting the social media domain have been defined in the literature as *social recommender systems* [15]. Recommender systems have first gained popularity via their usage in e-commerce applications to help users identify, from an overwhelming number of items, the products that best fit their personal preferences [17]. Indeed, recommender systems perform the role of virtual experts who are keenly aware of user preferences and tastes and correspondingly filter out a vast amount of irrelevant data to identify and recommend the most relevant products [18]. For example, a popular application of recommendation is movie recommender systems such as *Netflix* that help users select, based on their tastes and preferences, a movie to watch from a vast catalog of movies [17].

Over the last decade, several algorithms have been developed for recommender systems and make use of user feedback such as purchase histories, ratings, and reviews to compute the recommendations [18]. In the meantime, the pervasive use of social media has been generating massive data at an unprecedented rate where the information overload problem became increasingly critical for both social media users and researchers [14]. Indeed, social media data have three characteristics that pose challenges for research [19]: First, social media data are large. For example, user-generated content such as tweets, comments, posts, and reviews have contributed to the creation of the concept of *Big Data*. Second, social media data can be noisy. For example, spams and trivial tweets are abundant on *Twitter*. Third, data from online social networks are dynamic, so that updates over short periods of time are common and represent an important dimension to consider when dealing with social media data. To effectively overcome these three challenges, social recommender systems have been proven to be effective in mitigating the information overload problem and improving the quality of user experience, as well as positively impacting the success of social media [11]. Furthermore, artificial intelligence and machine learning techniques have recently gained increasing attention to handle the ever-growing data generated by social media [12]. Moreover, open access to data through APIs provides researchers with unprecedented amounts of information to improve and optimize the performance of machine learning techniques [19]. The advancement of machine learning itself relies on large datasets, and social media is an ideal data source for developing and testing new machine learning techniques for the academic and industry communities [13].

In this chapter, we first present and define social media, including a brief history and the main challenges due to the increasing amount of data. Then, we provide background on recommender systems and describe three main classical recommendation approaches: content-based filtering, collaborative filtering, and hybridization of the two types of filtering. Finally, we introduce and present the machine learning aspect, which is becoming increasingly important in the recommendation, as it overcomes the limitations of classical recommendation systems that fail to cope with the increasing data generated by social media applications and the unique specificity of their social structural information. In the machine learning aspect, we discuss the two main approaches used in the learning process: supervised learning such as classification and regression, and unsupervised learning such as clustering and dimensionality reduction.

The chapter is structured as follows: section 2.2 presents social media including a brief history and main challenges, section 2.3 provides background on recommender systems, section 2.4 introduces the machine learning aspect including supervised and unsupervised learning, and section 2.5 concludes the chapter.

2.2 Social media

In 2020, 84% of Internet users and 49% of the world population are active on social media[1]. On these social platforms, any user can post and share updates with individuals from his social network [11]. There are two main aspects to define social media, the sociological aspect and the technological aspect. First, from a sociological point of view, according to Wassermann and Faust [20], a social network is a set of relationships between social entities, individuals, or organizations. The contacts between these individuals can be, for example, friendships, collaborative relationships, professional relationships, etc. Fig. 2.1 represents a small social network in which the nodes represent social entities, natural (individuals) or legal persons (organizations), and the links represent the social connections between these entities. The links refer to the set of relationships that unite entities that belong to the same social group as social media users [21]. Indeed, social media users tend to have friend relationships and share their opinions with other groups of users [16]. Second, from a technological point of view, according to Kaplan and Haenlein [14], social media is the group of internet-based applications that build on the ideological and technological foundations of Web 2.0 and that allow the creation and exchange of user-generated content on the web. The most important user-related concepts on social media are as follows:

- The user's personal page, where content posted by the user is displayed
- The news feed, where the user is exposed to content posted in his social network
- Social relationships: friends, followers, followings, etc.
- Social interactions: click, comment, like, share, etc.
- Explicit profile: username, location, language, biography, explicit interests, etc.
- Implicit profile: browsing history, interaction history, implicit interests, etc.

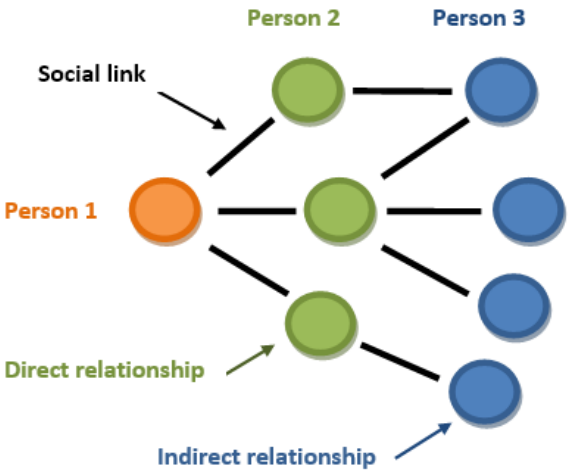

Fig. 2.1: Social network representation

Social media gives users an easy-to-use way to communicate and network with each other on an unprecedented scale and at rates unseen in traditional media [14]. The popularity of social media continues to grow exponentially, resulting in an evolution of social networks, blogs, location-based social networks, etc. There are many categories of social media including, but not limited to, social networking such as *Facebook* and *LinkedIn*, microblogging such as *Twitter*, photo sharing such as *Flickr* and *Instagram*, video sharing such as *YouTube*, and instant messaging such as *Skype* and *Yahoo! messenger* [14]. The first social media site was introduced by Geocities in 1994, which allowed users to create their own homepages [3]. Since then, many other social media sites have been introduced, each providing different services to millions of people. These social platforms form a virtual world where individuals, content entities, and interactions between individuals, between entities, between individuals and entities coexist [14]. Due to their ease of use and the many benefits they bring, social media platforms are experiencing a surprising increase in the number of users [11]. Furthermore, the huge amount of data and resources related to the social actors and their different interactions lead to information overload [3]. Hence, the increasing amount of data requires extensive filtering and recommendation processes to ensure that the data provided is of potential interest to users [16].

The vast amounts of user-generated content on social media can be mined and analyzed by understanding social norms and models of user behavior and combining

[1]https://wearesocial.com/digital-2020

them with the observations and measurements of this virtual world [12]. Indeed, social media mining is the process of representing, analyzing, and extracting relevant patterns from data in social media, resulting from social interactions [3]. Social media mining is an interdisciplinary field encompassing different techniques from computer science, data mining, machine learning, social network analysis, network science, sociology, ethnography, statistics, optimization, and mathematics [3]. This field faces many challenges such as the big data phenomenon, obtaining sufficient samples, finding relevant content, the noise removal fallacy, and the evaluation dilemma [14]. In addition, social media mining provides the necessary tools to mine this virtual world for interesting patterns. Some of the applications of social media analysis and mining are as follows [12]: (1) analyze information diffusion; (2) fake news and spam detection; (3) identify influential and expert users; (4) detect implicit or hidden communities in a social networking site; (5) opinion mining and sentiment analysis; (6) understand the social network evolution and changing entity relationships; (7) build and strengthen trust among users or between users and entities; and (8) develop recommendation systems for tasks ranging from buying specific products to content recommendations to friend suggestions. In this chapter, we focus on the recommendation and provide in the next section background on traditional recommender systems.

2.3 Recommender systems

Internet users make a variety of decisions on a daily basis, such as buying a product, purchasing a service, watching a movie, adding a friend on social media, reading a given content, etc. [15]. Due to the large amount of data, users face many options to choose from. Indeed, the many and diverse options, the pursuit of optimality, and the limited knowledge and time that each user has create a desire for external help [21]. Occasionally, users resort to search engines for recommendations [11]. However, the results in search engines are rarely tailored to the particular preferences of users and are query-dependent, i.e. independent of the individuals who search for these queries [21]. To overcome this problem, algorithms and applications have been proposed and developed to help users decide easily, rapidly, and more accurately. Indeed, the latter algorithms are designed to recommend personalized and individual-based items that are tailored to the preferences of each user, such that the same query issued by different users should result in different recommendations [16]. These algorithms are called *recommendation algorithms* or *recommender systems* [18].

Traditional recommender systems attempt to recommend items based on various signals from users, such as aggregated ratings of items, product purchase histories, and user interactions [18]. On the other hand, a social recommendation system makes use of the social network and related user profile information in addition to the traditional recommendation means [15]. Indeed, a social recommendation system is based on the hypothesis that people who are socially connected are more likely to share similar interests [16]. Furthermore, users can be easily influenced by the friends they trust and prefer their friend recommendations to random recommendations [21]. The objectives of social recommendation systems are to improve the quality of the recommendation and alleviate the problem of information overload on social media [15]. Examples of social recommendation systems include book recommendations based on reading lists of friends on *Amazon* and friend recommendations on *Twitter* and *Facebook*.

A recommender system consists of two basic entities: users and items such as articles, books, movies, etc., where users can provide their opinions about items via ratings [11]. Formally, we denote the users by $\mathcal{U} = \{u_1, u_2, \cdots, u_M\}$, where the number of users is $|\mathcal{U}| = M$, and denote the set of items being recommended by the system by $\mathcal{I} = \{i_1, i_2, \cdots, i_N\}$, with $|\mathcal{I}| = N$. A recommendation algorithm takes a set of users U and a set of items I and learns a function f such that [17]:

$$f : U \times I \to \mathbb{R} \tag{2.1}$$

In other words, the algorithm learns a function that assigns a real value to each user-item pair (u, i), where the value indicates how interested the user u is in the item i. Indeed, this value often denotes the rating given by the user u to the item i [11]. Note that the recommendation algorithm is not limited to item recommendations and can be generalized to recommending people and material such as ads or contents [16]. A typical recommendation engine processes data through the following three phases: collection, storing, and filtering [17]:

1. Collecting the data: collect explicit and implicit user data. The explicit data consist of data inputted directly by users, such as ratings and comments on items. On the other hand, the implicit data consist of the user's order history, browsing history, page views, clicks, etc.

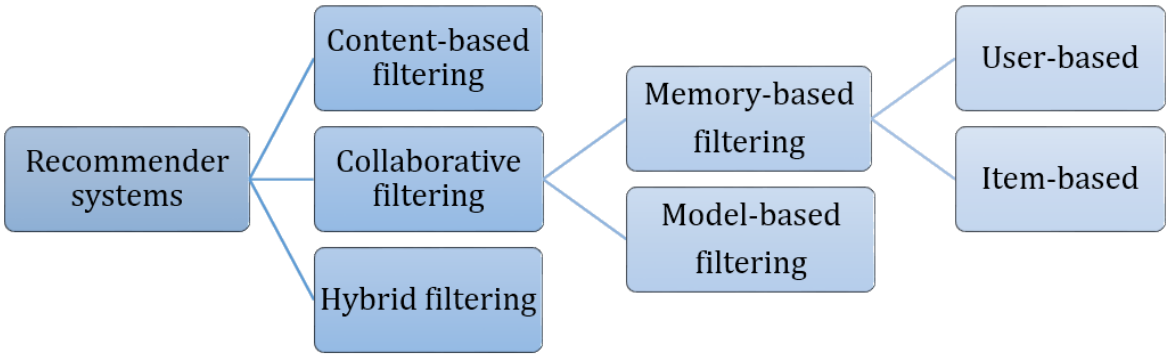

Fig. 2.2: Classical approaches in recommender systems

2. Storing the data: construct and store for users and items a vector, matrix, or model containing the information collected in the first step. The preferences known by users are generally characterized by their appreciation of the items already consulted.

3. Filtering the data: filter the data to get the relevant data required to provide useful recommendations to users. For this matter, a recommendation algorithm that suits the recommendation engine is used.

Classical recommendation algorithms have a long history on the web [18]. In recent years, with the emergence of social media sites, recommendation algorithms have been provided new information, such as user profile and friendship information, news feed updates, social interactions, etc. [16]. The classical recommendation approaches are defined as follows (see Fig. 2.2) [17]:

- Content-based filtering: based on the description of items similar to items already appreciated by the user [22].
- Collaborative filtering: based on the collective user ratings on items [23].
- Hybrid filtering: combines content-based and collaborative filtering approaches [24].

In the rest of this section, we review the three classical recommendation approaches mentioned above and discuss their advantages and disadvantages.

2.3.1 Content-based filtering

In content-based filtering, the system is based on the assumption that a user's interest should match the description of the items that are recommended [22]. In other words, the more similar the item's description to the user's interest, the higher the likelihood that the user will find the recommendation of that item interesting. Indeed, content-based recommender systems implement this idea by measuring the similarity between an item's description and the user's profile information. The higher the similarity, the higher the probability that the item is recommended [11]. To create a user profile, the system focuses on two types of information: the model of the user's preferences and the history of the user's interactions with the recommender system [25].

To formalize the content-based method, both user profiles and item descriptions are first represented by vectors using a set of k keywords [18]. After vectorization, the item j can be represented as a $k-$dimensional vector $I_j = \left(i_{j,1}, i_{j,2}, \ldots, i_{jk}\right)$ and the user i as the vector $U_i = \left(u_{i,1}, u_{i,2}, \ldots, u_{i,k}\right)$. To compute the similarity between the user i and the item j, the following Cosine similarity can be used between the two vectors U_i and I_j [17]:

$$\operatorname{sim}(U_i, I_j) = \cos(U_i, I_j) = \frac{\sum_{l=1}^{k} u_{i,l} i_{j,l}}{\sqrt{\sum_{l=1}^{k} u_{i,l}^2} \sqrt{\sum_{l=1}^{k} i_{j,l}^2}} \tag{2.2}$$

In content-based recommendation, the top-most similar items to a user are computed and then recommended in the order of similarity. Indeed, there are four main steps to recommend items in content-based approaches [17]:

1. The system gathers information about items. For example, in a movie recommender system, this would be the movie title, genre, actors, producers, etc.;

2. The user is asked to rate some items. For example, binary scales in terms of likes/dislikes or some numeric scale from 1 to 5 can be used for capturing the user's ratings;

3. The user's profile is built based on the information gathered in the first step and the rating provided in the second step. Different machine learning and information retrieval techniques can be used for this purpose. User profiles are updated as more information about user preferences is observed and are highly dependent on the learning method employed;

4. The system matches the content of unrated items with the active user's profile and assigns a relevance score to each item based on a measure of similarity.

Content-based filtering has been widely used in recommender systems thanks to its many advantages, some of which are mentioned in the following [17]:

- The model only exploits user interactions and content description of items to generate recommendations. Indeed, interactions made by other users are not needed such that items can be recommended based on their descriptions even if they have not been rated by any user. Therefore, the model can deliver recommendations in item cold-start scenarios.

- The model can capture the specific interests of a user even if the user has a unique taste and recommend niche items that few other users are interested in.

- It is possible to explain the results of the recommendations by providing the set of content similarity features that caused an item to appear in the recommendation list. These features increase the transparency of the system and indicate whether the user can trust the recommendations.

However, despite its advantages, content-based filtering suffers from several disadvantages, some of which are mentioned in the following [17]:

- The performance of the model is tied to the number and type of description features associated with items. Indeed, domain knowledge is often required, and enough information about items, which is not always available, should be gathered to discriminate items appropriately.

- The model recommends items similar to items that the user has previously interacted with and cannot provide different and unexpected suggestions. Indeed, content-based models suffer from overspecialization and are not able to provide novel or serendipitous recommendations.

- Building a consistent user profile requires gathering enough user interactions with items, making content-based filtering approaches unsuitable for cases where only a few interactions are available such as in new user cold-start scenarios.

Table 2.1: User-Item rating matrix

	Lion King	Aladdin	Mulan	Anastasia
Mohamed	3	0	3	3
Ahmed	5	4	0	2
Meriem	1	2	4	2
Aicha	3	?	1	0
Yacine	2	2	0	1

2.3.2 Collaborative filtering

In collaborative filtering, a user-item rating matrix is given where each entry is either unknown or is the rating assigned by the user to an item [23]. For example, Table 2.1 is a user-item rating matrix where ratings for some cartoons are known and unknown for others, indicated by question marks. For instance, on a review scale of 5, where 5 is the best and 0 is the worst, if an entry (i, j) in the user-item matrix is 4, that means that the user i liked the item j.

In collaborative filtering, the aim is to predict the missing ratings in the user-item matrix and recommend the items with the highest predicted ratings [18]. The prediction can be performed directly by using previous ratings in the matrix. This first approach is called *memory-based* collaborative filtering because it employs historical data available in the matrix [23]. Alternatively, an underlying model or hypothesis can be assumed to govern the way users rate items. The model can be approximated and learned from the data and then be used to predict other ratings. This second approach is called *model-based* collaborative filtering [23]. In the rest of the section, we describe each approach and discuss the advantages and disadvantages of collaborative filtering.

2.3.2.1 Memory-based filtering

Memory-based collaborative filtering consists of two methods such that one of the following assumptions, or both, are assumed to be true [23]:

- User-based filtering: users with similar previous ratings for items are likely to rate future items similarly

- Item-based filtering: items that have previously received similar ratings from users are likely to receive similar ratings from future users

In both cases, users or items collaboratively help filter out irrelevant content. To determine the similarity between users or items, two commonly used similarity measures are the Cosine similarity and the Pearson correlation [17]. Indeed, let $r_{u,i}$ denote the rating that the user u assigns to the item i, $\bar{r}_u$ the average rating for the user u, and $\bar{r}_i$ the average rating for the item i. The Cosine similarity between two users u and v is calculated using Equation 2.3 [17]. Note that the similarity between two items is calculated in the same way.

$$\text{sim}\,(U_u, U_v) = \cos\,(U_u, U_v) = \frac{U_u \cdot U_v}{\|U_u\|\,\|U_v\|} = \frac{\sum_i r_{u,i} r_{v,i}}{\sqrt{\sum_i r_{u,i}^2}\sqrt{\sum_i r_{v,i}^2}} \tag{2.3}$$

The Pearson correlation coefficient between two users is given by Equation 2.4 [17]. Note that the similarity between two items is calculated in the same way.

$$\text{sim}\,(U_u, U_v) = \frac{\sum_i (r_{u,i} - \bar{r}_u)\,(r_{v,i} - \bar{r}_v)}{\sqrt{\sum_i (r_{u,i} - \bar{r}_u)^2}\sqrt{\sum_i (r_{v,i} - \bar{r}_v)^2}} \tag{2.4}$$

In user-based filtering, the model finds similar users who have similar ratings for similar items, then the target user's rating for the item he has never rated is predicted. On the other hand, in item-based filtering, the model finds similar items to items the target user has already rated, then the target user's rating for the item is predicted. For example, to predict the rating of a user u on an item j, the most used measure for the prediction is the weighted sum given by Equation 2.5 [23]. The equation considers N_u, the most similar neighbours to the user u who have rated the item j. Note that memory-based filtering uses all available ratings to make predictions, so as the number of users and items increases, the complexity of processing becomes exponential [18].

$$r_{u,j} = \bar{r}_u + \frac{\sum_{v \in N_u} \text{sim}\,(U_u, U_v) * (r_{v,j} - \bar{r}_v)}{\sum_{v \in N_u} \text{sim}\,(U_u, U_v)} \tag{2.5}$$

2.3.2.2 Model-based filtering

Model-based filtering assumes that an underlying model governs the way users rate items [23]. The aim is to learn the model from the rating data and then use the model to predict the missing ratings in the user-item matrix. *Latent factor* models [26] are a state-of-the-art methodology for model-based collaborative filtering. The

basic assumption is that there exists an unknown low-dimensional representation of users and items where user-item affinity can be modeled accurately [23]. For example, the rating that a user gives to a movie might be assumed to depend on a few implicit factors such as the user's taste across various movie genres. Matrix factorization techniques [27] are a class of widely successful *latent factor* models that attempt to find weighted low-rank approximations to the user-item matrix, where weights are used to predict missing entries. There is a large family of matrix factorization models based on the choice of the loss function to measure approximation quality, regularization terms to avoid overfitting, and other domain-dependent formulations [17].

Among a variety of matrix factorization techniques, we focus on a well-established technique based on Singular Value Decomposition (SVD) [28]. SVD is a linear algebra technique that, given a real matrix $X \in \mathbb{R}^{m \times n}$ $m \geq n$, factorizes it into three matrices U, Σ, and V^T as given by Equation 2.6 [28].

$$X = U \Sigma V^T \tag{2.6}$$

Where $U \in \mathbb{R}^{m \times m}$ and $V \in \mathbb{R}^{n \times n}$ are orthogonal matrices and $\Sigma \in \mathbb{R}^{m \times n}$ is a diagonal matrix [28]. The product of these three matrices is equivalent to the original matrix X, which means that no information is lost.

Then, the best rank-k approximation of the matrix can be computed by calculating the SVD of the matrix, taking the first k columns of U, truncating Σ to the first k entries, and finally taking the first k rows of V^T [17].

Finally, given the user-item matrix X, its noise can be removed by computing X_k from X and obtaining the new k-dimensional user space U_k or the k-dimensional item space V_k^T [17]. In this way, the most similar user or item neighbors can be computed based on similarity distances in this k-dimensional space. The similarity in the k-dimensional space can be computed using the Cosine similarity or the Pearson correlation [17] (see Equations 2.3 and 2.4).

2.3.2.3 Advantages and disadvantages

Collaborative filtering provides many advantages over content-based filtering, some of which are as follows [17]:

- The model does not require domain knowledge as required in content-based recommender systems. Indeed, collaborative filtering generates recommendations by only considering the implicit feedback of users in the user-item matrix.
- The model can make different recommendations outside the preferences of individual users to help them discover new interests. For example, a user who loves action movies can also enjoy watching a romantic movie that similar users have enjoyed.
- The model captures more information about user preferences over time, which results in improved recommendations.

However, despite its advantages, collaborative filtering suffers from several disadvantages, some of which are as follows [17]:

- The performance of the model suffers under user and item cold-start problems where new users and items have no ratings. The new item problem is known as the *early-rated problem* since the first user to rate the item gets a little reward.
- The percentage of ratings assigned by users to items is very small compared to the percentage of ratings the system has to predict. Therefore, the prediction accuracy of the model suffers in this case from data sparsity.
- There can be users in the system with unusual or unique tastes compared to the rest of the community. In this case, the model would produce poor recommendations for these users.

2.3.3 Hybrid filtering

Since content-based and collaborative filtering approaches rely on different input sources to make recommendations, each one of them has its advantages and disadvantages. Therefore, hybrid methods have been proposed to get the best of both approaches [24]. Indeed, hybrid methods assume that various sources of input are available at the same time, which allows the use of different recommendation approaches in one framework and avoids the limitations of each approach when used separately [18]. On the other hand, competitions such as the *Netflix Prize* highlighted that the best results are often achieved when different recommendation algorithms are combined in a single model [29]. Hybrid methods can take various forms, and an existing classification covering the main trends of hybrid recommendation is presented in the following [25]:

- Combine separate recommendations by several recommendation algorithms to provide a single recommendation using methods such as weighted linear combinations and voting schemes.
- Incorporate content-based characteristics into collaborative filtering approaches. For example, user-based filtering can be adapted to compute similarities using content-based user profiles.
- Incorporate collaborative characteristics into content-based filtering approaches. For example, collaborative filtering can be applied to a group of content-based user profiles for text recommendation.
- Develop a unifying model that incorporates content-based and collaborative characteristics. In this scope, several approaches have been proposed, such as a unified probabilistic method for combining collaborative and content-based recommendations.

In this section, we have seen that the choice of one classical recommendation approach is strongly linked to the problem being addressed and the data available to the recommendation system. In the next section, we introduce the machine learning aspect, which is becoming increasingly important in the recommendation. Indeed, machine learning can overcome the limitations of classical recommendation systems that fail to cope with the increasing data generated by social media and the unique specificity of their social structural information (social relationships, social interactions, etc.) compared to traditional items such as articles, books, movies, etc.

2.4 Machine learning

In recent years, with the evolution of the field of recommender systems and the increasing popularity of machine learning and social media, researchers have studied the use of machine learning algorithms to provide social media users with better recommendations [13]. Indeed, classical recommendation approaches are unsuitable to analyze the enormous amount of real-time social data produced by social media [12]. In contrast, machine learning approaches can play an important role in overcoming this problem as the advancement of the machine learning field itself relies on large datasets, and social media is an ideal data source for developing and testing new machine learning techniques [13]. For example, machine learning algorithms can

process the historical data about user interactions and learn a model that can compile a ranked list of all items available for recommendation for each user based on the information encoded in their profile [19]. The highly ranked items can then be recommended to the corresponding user based on how likely this user will consume these items.

Machine learning is an application of artificial intelligence (AI) that provides systems the ability to automatically learn and improve from data and experience without being explicitly programmed [30]. Learning is known as the process of knowledge acquisition. Humans naturally learn from experience because of their ability to reason. In contrast, computers do not learn by reasoning but with algorithms [31]. Machine Learning simulates human learning and allows computers to identify and acquire knowledge from the real world and improve the performance of some tasks based on this new knowledge. More formally, Carbonell et al. [32] define machine learning as follows: "A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if the performance of the program at tasks in T, as measured by P, improves with experience E". Although the first concepts of machine learning originated in the 1950s, it was studied as a separate field in the 1990s [30]. Today, machine learning algorithms are used in several areas besides computer science, including business, advertising, medicine, etc. [33].

In machine learning, the features are the input variables used by the machine learning model to perform a specific task [31]. The features are usually numerical. For example, in spam detection algorithms, the features may include the presence or absence of some email headers, the email structure, the language, the frequency of specific terms, the grammatical correctness of the text, etc. [34]. Feature engineering is the process of using domain knowledge to create input variables that help machine learning algorithms perform better [35]. Correctly performed, feature engineering can help increase the predictive power of the model. Indeed, feature engineering is one of the most important steps in creating a good machine learning model [34].

In the literature, different machine learning algorithms have been proposed to solve a wide range of problems [30]. However, according to the *no free lunch* theorem [36], no single machine learning algorithm is better than all the others on all problems. Indeed, some algorithms perform very well in some specific problems, but no algorithm performs well in all problems. Moreover, each algorithm has its advantages and dis-

advantages such as flexibility, complexity, interpretability, overfitting tendency, times for learning and predicting, tolerance to a large number of features, minimal required data, number of hyperparameters[2], etc. [34]. Therefore, choosing the right algorithm depends mainly on the problem. For example, the category such as image recognition, sentiment analysis, text classification, etc., the type such as classification, regression, clustering, etc., the number of features, the size of the data, etc. [33]. Most of the time, it is common to try different well-known machine learning algorithms and select the best one that works best for a particular problem by a validation technique [33].

Machine learning algorithms can be classified into two main classes based on the approach used in the learning process: supervised and unsupervised learning. In the rest of this section, we discuss each approach in more general terms, while we discuss their use in the field of ranking news feed updates in the chapter 3.

2.4.1 Supervised learning

In supervised learning algorithms, the class attribute values for the dataset are known before running the algorithm. This dataset is called *labeled data* or *training data* [31]. In the training data, the instances are tuples in the format (x, y), where x is an input feature vector, and y is the output class attribute, commonly a scalar [30]. The supervised learning approach builds a model that maps an input x to an output y. The task is roughly to find a mapping function $m(.)$ such that $m(x) = y$ [33]. The model is also given an unlabeled dataset called *test data*, in which instances are in the form $(x, ?)$ where y values are unknown [31]. Given the function $m(.)$ learned from the training data and an input feature vector x of an unlabeled instance, the model can compute $m(\mathbf{x})$, which is the prediction of the label for the unlabeled instance [30].

The supervised learning process is described in Fig. 2.3. The process starts with labeling data in the training set, where both features and the desired output of the labels are known. A supervised learning algorithm is then run on the training set in a process known as *induction* [34]. In the induction process, the model is trained and generated such that the resulted model learns to map the input feature values to the output corresponding to the class attribute values. Then starts the deduction process [34]. In the latter, the model is used on a test set to predict the corresponding

[2]The "knobs" to tweak during successive runs of training a model.

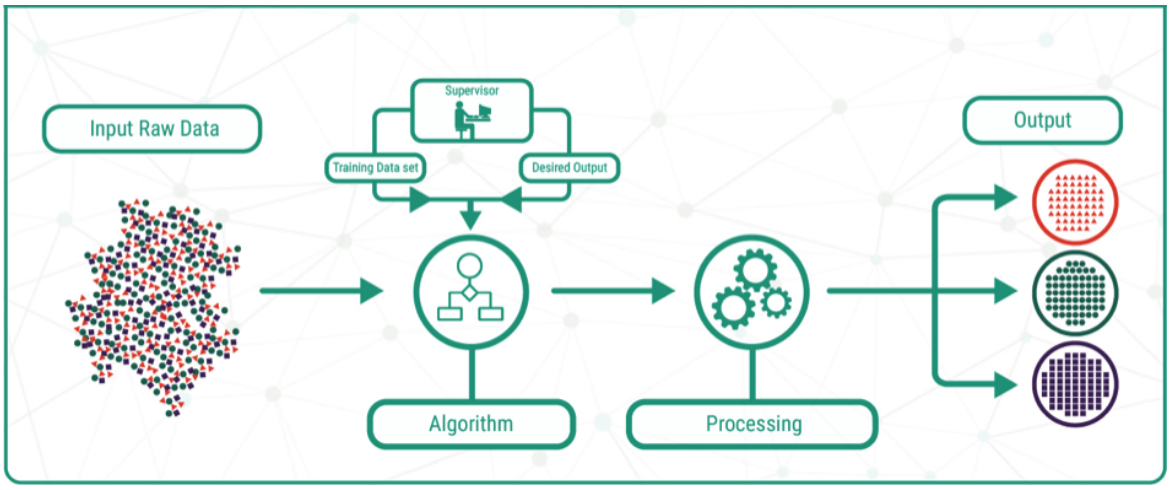

Fig. 2.3: The supervised learning process [37]

class attribute values, which are initially unknown. For example, consider the task of detecting spam emails. A set of emails is given in the training data where users have manually identified and labeled spam versus non-spam emails. The task is to use a set of features such as words in the email (x) to identify the spam/non-spam status (y) of unlabeled emails in the test data. In this case, $y = \{\text{spam, non-spam}\}$.

According to the type of the output class attribute, supervised learning can be divided into classification and regression problems. Indeed, when the class attribute values are discrete, it is called *classification*. On the other hand, when the class attribute values are continuous, it is called *regression*. In the rest of this section, we briefly discuss both classification and regression methods.

2.4.1.1 Classification

In classification, the class attribute values are discrete such that the computer program is asked to specify which of the k categories some input instances belong to [38]. For example, in classification in Fig. 2.4[3], the black line represents a classification function that classifies input instances into two categories, such that k=2, which corresponds to a binary classification problem. To solve a classification task, the machine learning algorithm is asked to produce a function $f : \mathbb{R}^n \rightarrow \{1, \ldots, k\}$ [30]. Indeed, when $y = f(x)$, the model assigns an input instance described by a vector x to a category identified by the numeric code y [31]. There are other variants of the classification task, for example,

[3]https://www.javatpoint.com/regression-vs-classification-in-machine-learning

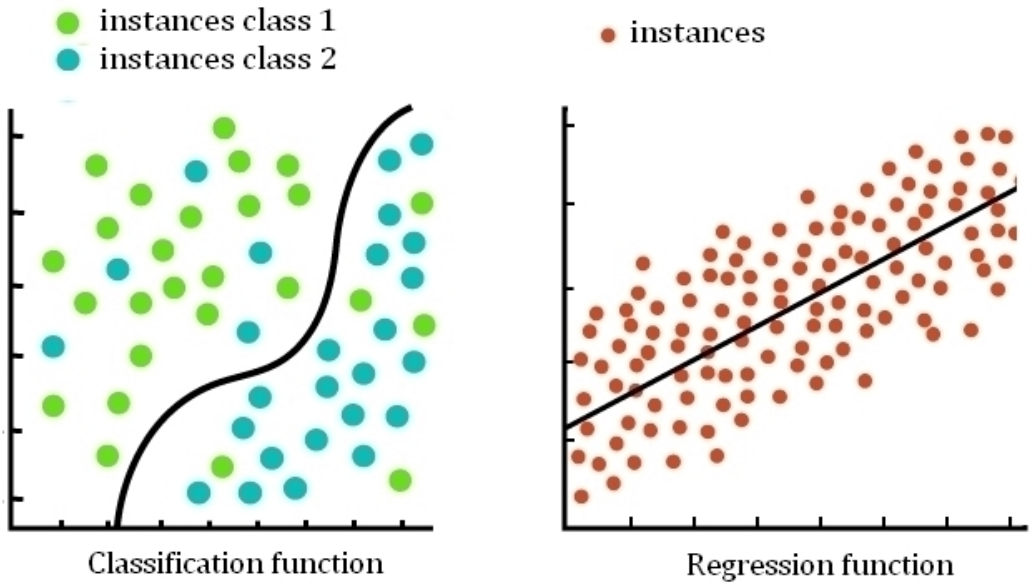

Fig. 2.4: Classification vs. Regression

where the function f outputs a probability distribution over classes [30]. An example of a classification task is object recognition, where the input is an image described as a set of pixel brightness values, and the output is a numeric code identifying the object in the image, e.g. 0 for a cat and 1 for a dog [33]. Another example is the *Willow Garage PR2* robot, which acts as a waiter that recognizes different kinds of drinks and deliver them to people on command [39]. There are several algorithms that are commonly used to solve classification problems [38]. Later in the chapter 5, we describe some important classification algorithms such as *logistic regression* [40], *decision trees* [41], *Naive Bayes* [42], *random forest* [43], *artificial neural networks* [33], etc.

2.4.1.2 Regression

In regression, the class attribute values are real numbers. The regression analysis is used to understand the relationship between two or more variables of interest and examines the influences of one or more independent variables on a dependent variable [34]. Indeed, regression analysis allows for a better understanding of the specific ways a dependent variable is affected by an independent variable. In short, the regression answers how the dependent variable changes when varying one of the independent variables, while other independent variables remain unchanged [30]. For example, the stock market value of a company can be predicted given information about the company, such that the stock value is the class attribute and the information about

the company are the features. The stock market value is continuous, which means that a regression algorithm must be used to predict it. The input to the regression method is a dataset where attributes are represented using $x_1, x_2, \ldots, x_m$, also known as *regressors*, and the class attribute is represented using Y, also known as the *dependent variable*, where the class attribute is a real number [34]. The goal is to find the relationship between the class attribute Y and the vector $X = (x_1, x_2, \ldots, x_m)$. For example, Fig. 2.4 visualizes a linear regression function across a specified dataset[3]. The black line represents the linear regression function and expresses the rate of change among the data points. There are several algorithms that are commonly used to solve regression tasks [30]. Later in the chapter 5, we describe some important regression algorithms such as *decision trees* [41], *artificial neural networks* [33], *Support Vector Machine* [44], *Gradient Boosting* [45], etc.

2.4.2 Unsupervised learning

In unsupervised learning algorithms, a dataset is provided without labels and without a training set, and a model is used to draw inferences, discover hidden patterns, and learn useful properties of the structure of the dataset [46]. This is in contrast to supervised learning techniques, such as classification or regression, where a model is given a training set of inputs and a set of observations and must learn a mapping from the inputs to the observations. Unsupervised learning is commonly used to find interesting patterns and draw conclusions from the unlabeled data when the output labels are not known in advance [30].

The unsupervised learning process is described in Fig. 2.5. The process starts with unlabeled data and no training set, where the features are known, and the desired output of the labels is unknown. Then, an unsupervised learning algorithm is run on the input row data to identify patterns within the data and categorize the input objects based on the patterns that the algorithm has identified [46]. The algorithm analyzes the underlying structure of the dataset by extracting useful information or features [30]. Indeed, unsupervised learning algorithms are expected to develop specific outputs from the unstructured inputs by looking for relationships between each sample or input object [34]. For example, suppose that a machine learning algorithm has access to user profile information on a social media. By using an unsupervised learning approach, the algorithm can separate users into personality categories, such as outgoing and reserved, allowing the social media company to target advertising more directly at

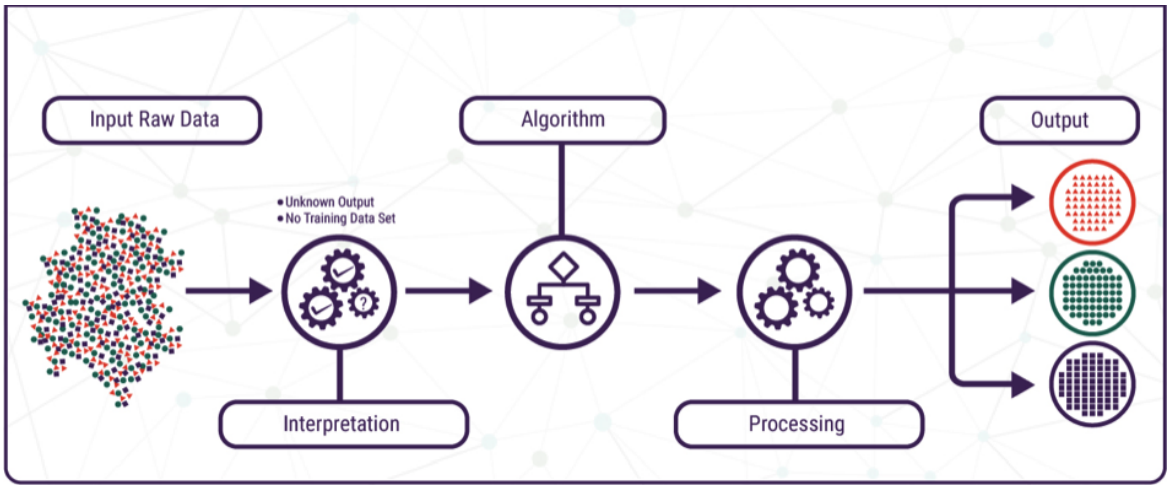

Fig. 2.5: The unsupervised learning process [37]

specific groups of users. Unsupervised learning tasks typically involve clustering and dimensionality reduction, which we discuss in the following.

2.4.2.1 Clustering

The most common unsupervised learning method is clustering analysis, which is used for exploratory data analysis to find hidden patterns or grouping in data [47]. Typically, a clustering algorithm requires a distance measure between instances such that the input instances are put into different clusters based on their distance to other instances [35]. The most popular distance measure for continuous features is the Euclidean distance, which is calculated as follows:

$$\begin{aligned} d(X,Y) &= \sqrt{(x_1-y_1)^2+(x_2-y_2)^2+\cdots+(x_n-y_n)^2} \\ &= \sqrt{\sum_{i=1}^{n}(x_i-y_i)^2} \end{aligned} \tag{2.7}$$

Where $X=(x_1,x_2,\ldots,x_n)$ and $Y=(y_1,y_2,\ldots,y_n)$ are n-dimensional feature vectors in $\mathbb{R}^n$. The instances are grouped into clusters using the selected distance measure. The clusters are usually represented by compact and abstract notations such as cluster centroids [47]. Finally, the clusters are evaluated such that the goal is to obtain high intra-cluster similarity and low inter-cluster similarity [34]. In other words, the objects in the same cluster should be more similar than the objects in the other clusters. There are many types of clustering algorithms depending on the methodology used to partition the data [47]. In this section, we discuss partitional clustering algorithms,

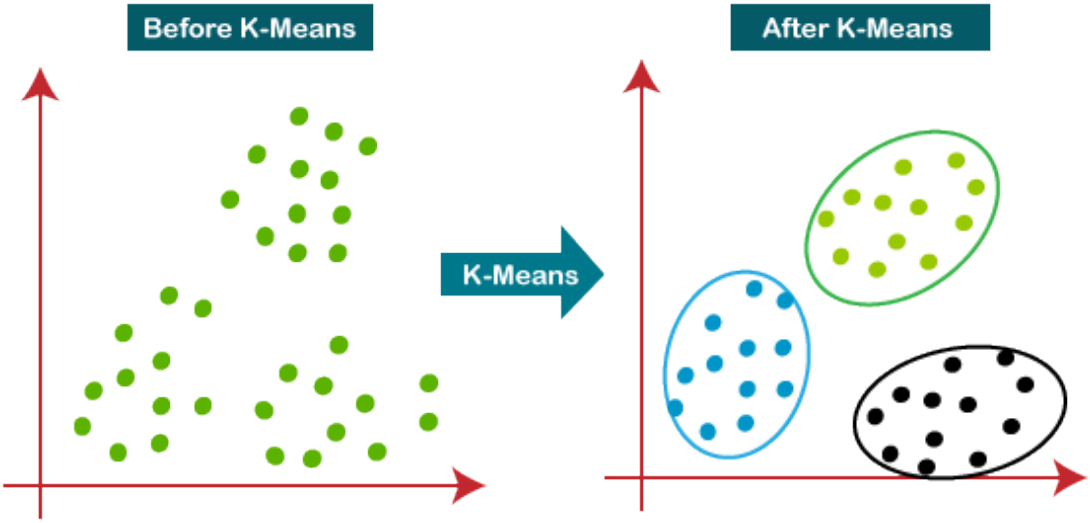

Fig. 2.6: K-means on a sample dataset, with $k = 3$

which are the most frequently used algorithms [34].

In partitional clustering, each instance is assigned to a cluster exactly once, and no instance remains unassigned to clusters [47]. K-means [48] is a well-known example of a partitional algorithm. The output of the k-means algorithm on a sample dataset with $k = 3$ is shown in Fig. 2.6[4]. In this figure, the dataset has two features, such that the instances are visualized with colored points in a two-dimensional space. The colors represent the clusters to which the instances belong. The k-means procedure is described as follows [48]: "First, the algorithm starts with k initial centroids, which are randomly chosen instances from the dataset. The initial random instances form the initial set of k clusters. Then, each instance is assigned to a cluster based on its distance to the centroid of each cluster. The calculation of distances from instances to centroids depends on the choice of the distance measure. The Euclidean distance is the most widely used distance measure (see Equation 2.7). After assigning all instances to a cluster, the centroids are recomputed by taking the average of all instances inside the clusters. Finally, this procedure is repeated using the newly computed centroids until convergence". The most common criterion to determine convergence is to check whether centroids are no longer changing [30]. This is equivalent to a stabilization of the clustering assignments of instances. In practice, the algorithm execution can be stopped when the Euclidean distance between the centroids in two consecutive

[4]https://www.javatpoint.com/k-means-clustering-algorithm-in-machine-learning

steps is bounded above by a small positive value ϵ [34]. K-means implementations try to minimize an objective function. A well-known objective function is given by the following squared distance error [48]:

$$\sum_{i=1}^{k}\sum_{j=1}^{n(i)}\left\|x_j^j - c_i\right\|^2 \tag{2.8}$$

Where x_j^i is the jth instance of the cluster i, $n(i)$ is the number of instances in the cluster i, and c_i is the centroid of the cluster i [48]. The process stops when the difference between the objective function values of two consecutive iterations of the k-means algorithm is bounded by a small value ϵ. Note that k-means is highly sensitive to the initial k centroids and that different clustering results can be obtained on a single dataset depending on the initial k centroids [30]. This well-known problem can be addressed by running k-means multiple times and selecting the clustering assignment that is observed most often or is more desirable based on an objective function, such as the squared error [34].

2.4.2.2 Dimensionality reduction

Dimensionality reduction is the transformation of data from a high-dimensional space into a low-dimensional space that retains some meaningful properties of the original dataset, ideally close to its intrinsic dimension [49]. In machine learning, dimensionality reduction, also known as *feature reduction*, is the process of reducing the number of features in a heavy resource computation without losing important information [33]. Indeed, reducing the number of features means that the number of variables is reduced, making the learning task easier and faster. Feature reduction is divided into two processes: (1) feature selection, which filters irrelevant or redundant features in the dataset; and (2) feature extraction, which creates a smaller set of features that captures the most useful information in the dataset [35]. The most popular dimensionality reduction techniques are *Generalized Discriminant Analysis*, *Autoencoders*, *Non-Negative Matrix Factorization*, and *Principal Component Analysis* [49].

Dimensionality reduction removes multicollinearity in large datasets resulting in improvement of the machine learning model in use [30]. Another benefit of dimensionality reduction is that it makes data easier to visualize graphically, particularly when the dataset is reduced to two or three dimensions [49]. Furthermore, an interesting problem that feature reduction can help with is called the *curse of dimensionality*,

which refers to a group of phenomena in which a problem may have so many dimensions that the data becomes sparse [30]. Feature reduction is used to decrease the number of dimensions, making the data less sparse and more statistically significant for machine learning algorithms. For example, in the MNIST image dataset[5], the pixels on the image borders are often white, so it is possible to drop these pixels from the training set without losing much information [35]. Moreover, two neighboring pixels are often highly correlated. For example, two pixels can be merged into a single pixel with no loss of information by taking the mean of the two-pixel intensities [35].

The most popular dimensionality reduction algorithm is Principal Component Analysis (PCA) [50]. The algorithm performs a linear mapping of the data to a lower-dimensional space in which the data variance is maximized [30]. Indeed, PCA first identifies the hyperplane that lies closest to the dataset and then projects the data onto it [35]. PCA learns a representation of data, which is based on two of the following criteria: (1) a representation that has a lower dimensionality than the original input; and (2) a representation whose elements have no linear correlation with each other [50]. This is a first step toward the criterion of learning representations, whose elements are statistically independent. To achieve full independence, a representation learning algorithm must also remove the nonlinear relationships between variables [30].

2.5 Conclusion

In this chapter, we first presented and defined social media, including a brief history and the main challenges due to the increasing amount of data. Then, we provided background on recommender systems, which filter out a large mass of information to help users find interesting items based on their tastes and preferences. Afterward, we presented and described three main classical recommendation approaches: content-based filtering, collaborative filtering, and hybrid filtering. Following this study, it turns out that each approach has advantages but also disadvantages. Therefore, the choice of one recommendation approach is strongly linked to the problem being addressed and the data available to the recommendation system. Finally, we introduced and presented the machine learning aspect, which is becoming increasingly important in the recommendation, as it overcomes the limitations of classical recommendation systems that fail to cope with the increasing data generated by social media and the unique

[5]http://yann.lecun.com/exdb/mnist/

specificity of their social structural information. In the machine learning aspect, we discussed the two main approaches used in the learning process: supervised learning such as classification and regression, and unsupervised learning such as clustering and dimensionality reduction.

In the next chapter, we introduce the field of ranking news feed updates on social media and make a state-of-the-art of the existing approaches to show their advantages, their limitations, and identify open research issues. The survey will then allow us to propose several machine-learning-based models that address the limitations of the approaches proposed in the literature.

Chapter 3

Ranking news feed updates (RNFU)

3.1 Introduction

Social media such as *Facebook* and *Twitter* are used by hundreds of millions of users worldwide. These social platforms have experienced unprecedented popularity in the past decade and have contributed significantly to increase levels of user-generated content that have been fueling this growth [51]. Indeed, social media have become rich and diverse sources of information that compete with and complement traditional search engines in the diffusion of information [52]. Social media are designed to allow any user to post and share content with individuals from his social network [53]. Due to the large amount of data, users find themselves overwhelmed by updates displayed chronologically in their news feed[1], from most recent to least recent [54]. For example, there are about 1500 new updates every day in the news feed of a typical *Facebook* user[2]. Moreover, several research works have shown that most updates are considered irrelevant [55]. For example, Paek et al. [56] asked 24 *Facebook* users from their *Microsoft* organization to assign relevance scores to news feed updates. The overall average score was close to 0. Hence, large data volume and irrelevance make it difficult for users to catch up with the relevant updates in their news feed [4], especially for users who have a great number of social relationships [52]. For example, it is estimated that a standard *Facebook* user is likely to miss some relevant updates in the news feed even if the user spends an average of 55 minutes a day on the social media platform [56].

[1] A list that allows users to follow updates about individuals from their social network.
[2] https://longform.org/posts/who-controls-your-facebook-feed

In several research approaches, ranking news feed updates in descending relevance order has been achieved based on the prediction of a relevance score between a recipient user and a new update in the news feed [9]. A state-of-the-art covering three different aspects has been carried out in [1]. The first aspect focuses on users who produce information, users who read and benefit from this information, and the links between the two. The second aspect focuses on the content of the information. The third aspect concerns the structure of the underlying social network of users and the information they produce. However, the work does not cover four important criteria: (1) the features that may influence the relevance of updates to recipient users; (2) the relevance prediction models; (3) the methods used to obtain training and evaluation data; and (4) the target evaluation platforms. Based on these four criteria, and in order to complete the state-of-the-art carried out in [1], the objective of this chapter is to make a state-of-the-art of the existing approaches in the field of ranking news feed updates, to show their advantages, their limitations, and identify open research issues. The survey will then allow us to propose several machine-learning-based approaches that address the limitations of those proposed in the literature. This leads us to formulate the following questions: what are the features that may influence the relevance of updates to users on the one hand? On the other hand, from these features, what model to use to predict the relevance of a new update in the news feed? Moreover, how to train and evaluate the prediction model given that users do not explicitly provide relevance scores for news feed updates? Finally, what social media platforms have been targeted by existing work? Studying these questions is the subject of this chapter.

In this chapter, we first provide background on ranking news feed updates on social media, including defining news feeds, presenting statistics on data volume and irrelevance that confirm the need for ranking, and formalizing the ranking process. Then, we discuss work carried out in the field of ranking and predicting the relevance of news feed updates in both industrial and academic communities. Afterward, we analyze and compare the research works and expose their advantages and limitations according to four main criteria: (1) the features that may influence relevance; (2) the relevance prediction models; (3) the training and evaluation methods; and (4) the evaluation platforms. Finally, we identify open research issues to which we will make contributions.

The chapter is structured as follows: section 3.2 provides background on ranking

news feed updates on social media, section 3.3 discusses work carried out in this area, section 3.4 provides an analysis of existing work, section 3.5 discusses open research issues, and section 3.6 concludes the chapter and proposes future work.

3.2 Background

In this section, we provide background information on ranking news feed updates on social media, including defining news feeds, exposing statistics that confirm the need for ranking, and outlining the ranking process.

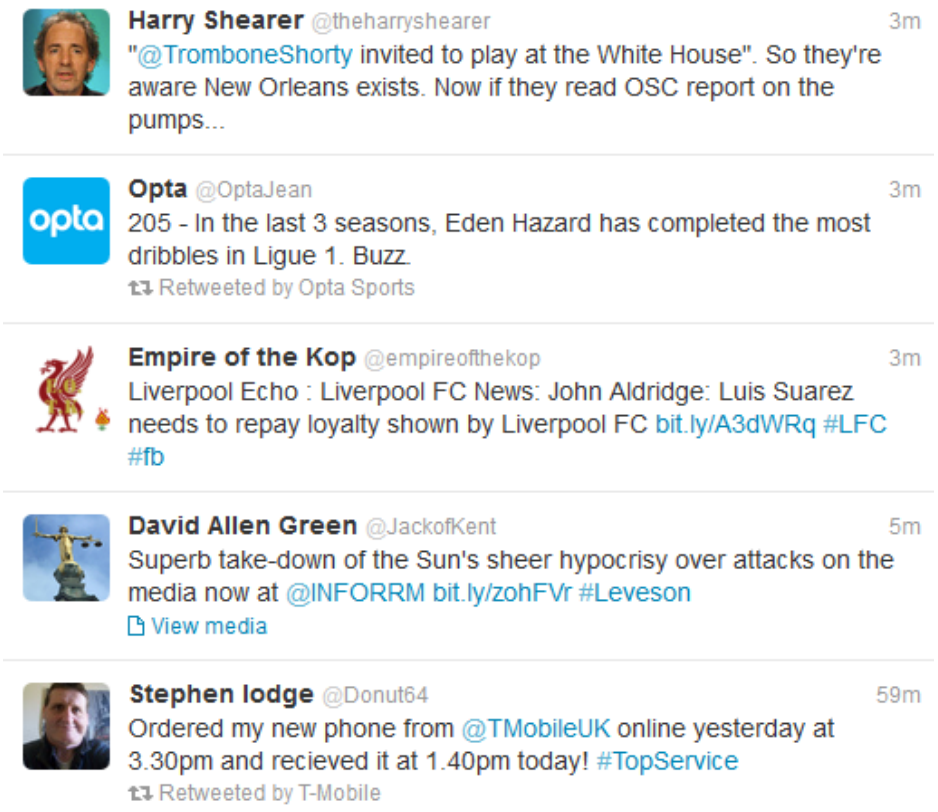

Fig. 3.1: A user's news feed on *Twitter*

3.2.1 News feed

According to Rader and Gray [57], and as shown in Fig. 3.1, the news feed is a list of information (posts, tweets, status updates, etc.) that allows a user to follow updates about individuals from his social network. Indeed, the news feed is the primary feature

through which users are exposed to content posted on social media. It includes text status updates, photos, videos, etc. [58]. For example, on a general public social media such as *Facebook* or *Twitter*, a user's news feed consists of updates about friends, family members, pages the user has subscribed to, etc. [21]. On the other hand, on a professional or academic social media such as *LinkedIn* or *ResearchGate*, a user's news feed consists of updates about contacts, colleagues, classmates, etc. [21].

Fig. 3.2: News feed update posted by *Elon Musk* on *Twitter*

On most social media, and as shown in Fig. 3.1, the news feed is displayed in reverse chronological order from the most recent to the least recent update [2]. Usually, news feeds do not support a keyword-based search for specific information needs, which makes it difficult for recipient users to find relevant updates on topics of interest to them [1]. Hence, the main disadvantage of the chronological news feed is that a user has to read and browse through a large number of updates to be sure not to miss relevant content [5]. Recipient users can unfriend, unfollow, or mute undesired connections from their social network who flood their news feed with irrelevant content. However,

the manual creation of rules and filters requires time and effort and only provides rigid options that may turn off relevant updates from some users [1].

Fig. 3.2 shows the main components of a news feed update [58, 1]: (1) an author who posted the update; (2) a set of recipient users who can read and interact with the update (click, comment, like, share, etc.); (3) a textual and/or multimedia content; (4) a publication date; (5) mentions, or names preceded by at-sign "@", which represent links to other users of the social media; (6) hashtags, or metadata markers preceded by a hashtag "#", which describe the topic of the update; and (7) URLs to websites or articles.

3.2.2 Statistics

In 2020, 84% of Internet users and 49% of the world population are active on social media[3]. On these social platforms, any user can post and share updates with individuals from his online social network (text messages, photos, videos, etc.) [53]. The increasing popularity of social media, the ease of sharing content by users, and the expanding number of social relationships have contributed to flooding the news feed concept [1]. Indeed, some statistics on news feeds may confirm the need for ranking, in particular, statistics on the large volume of data and its irrelevance.

Volume The volume of user-generated content in news feeds is overwhelming and constantly growing, and while time spent on social media sites has increased, the flood of incoming updates still greatly exceeds the capacity of information that any user can deal with. For example, there are about 1500 new updates every day in the news feed of a standard *Facebook* user, and this can go up to 10,000 updates for users who have hundreds of social relationships[4]. Because of the large volume of data on the photo and video-sharing social media *Instagram*, users only see 30% of the updates in their news feed[5]. In [59], 587 *Twitter* users answered a UK-based survey. The results showed that 66.3% of users feel that they cannot keep up with the large flow of updates in their news feed. In [60], following a survey of 56 *Twitter* users from within *Microsoft*, the results indicated that users have too many news feed updates. Kuang et al. [4] asked

[3]https://wearesocial.com/digital-2020
[4]https://longform.org/posts/who-controls-your-facebook-feed
[5]https://www.nytimes.com/2016/03/31/technology/instagram-is-changing-its-feed-but-calm-down-not-yet.html

1048 volunteers who are relatively active on the Chinese social media *Sina Weibo* to participate in their experiments. Statistics showed that users receive in their news feed an average of 56 newly posted updates per hour.

Irrelevance A number of research studies have shown that the incoming news feed updates are of marginal or no interest to recipient users. For example, in order to highlight the irrelevance of updates on *Facebook*, Paek et al. [56] asked 24 active users from their *Microsoft* organization to assign relevance scores to news feed updates. Similarly, Alonso et al. [61] asked 39 *Twitter* users, recruited from a crowdsourcing platform that specializes in relevance judgment, to assign relevance scores to more than 2000 updates. In both cases, the overall average relevance score was close to 0. In [59], 587 *Twitter* users answered a UK-based survey. The results showed that 70.4% of users have trouble finding the relevant updates in their news feed. In [60], following a survey of 56 *Twitter* users from within*Microsoft*, the results indicate that users lose the most relevant updates in a news feed of thousands of less useful updates. On *LinkedIn*, Agarwal et al. [8] assert that the chronological news feed leads to a recent but not necessarily relevant feed. Indeed, the authors ran an online test comparing the chronological news feed with a relevance-based feed and found the click-through rate[6] of the relevance-based news feed 43% higher than that of the chronological feed.

3.2.3 Ranking process

According to Shen et al. [5], ranking news feed updates on social media involves sorting and displaying in descending relevance order the news feed updates of each user. Nonetheless, we note that other terms can be used to refer to the ranking process, such as reordering, recommendation, personalization, etc [1]. Indeed, as reported by Berkovsky and Freyne [1], ranking news feed updates can be considered as either a top-K recommendation or a re-ranking problem. Unlike the chronological news feed, the ranking process is done such that the most relevant updates are found at the top of the news feed and the least relevant at the bottom [8].

Berkovsky and Freyne [1] propose the following formalization of the problem of ranking news feed updates. Let $\boldsymbol{F(u)}$ denotes all the updates unread by the recipient user $\boldsymbol{u}$ that can be included in the news feed. The ranking process implies selecting

[6]The ratio of users who click on an item to the number of total users who view it.

and displaying a subset $K(u) \in F(u)$, such that $|K(u)| \ll |F(u)|$, that corresponds to the most relevant updates to u. The ranking procedure involves three steps:

1. predict and assign a relevance score to each new update $i \in F(u)$;
2. select and display, in descending relevance order, the $|K(u)|$ updates with the highest relevance scores at the top of the news feed;
3. save or delete the remaining $F(u) \setminus K(u)$ updates.

The rest of this chapter focuses on the first step of the ranking process, which is the most important. Fig. 3.3 describes the primary technique used to predict and assign a relevance score to an update $i \in F(u)$. This technique is based on a prediction model that uses as input a set of features that may influence the relevance of the update i to the recipient user u, to output a corresponding relevance score $R(i,u)$ that measures the relevance of i to u. Note that i is posted by an author user $u' \in A(u)$, such that $A(u)$ is the set of users who belong to the social network of u. In the next section, we discuss work carried out to rank and predict the relevance of news feed updates.

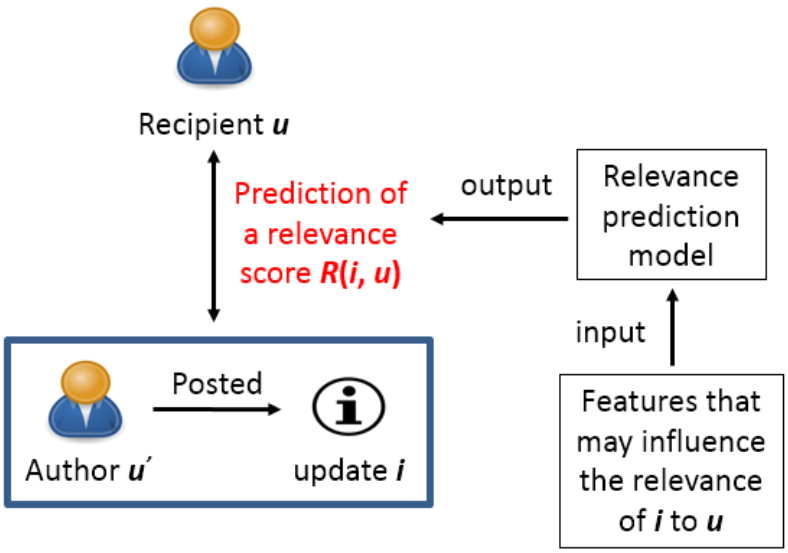

Fig. 3.3: Prediction of a relevance score

3.3 Related work

Ranking news feed updates is being actively studied in both industrial and academic communities. In this section, we discuss work carried out in each community.

3.3.1 Industrial community

In the industrial community, *Facebook*, *Twitter*, and *LinkedIn* are making efforts to rank and display news feed updates in descending relevance order. However, their approaches are most often undisclosed due to commercial sensitivity and competition between companies [1]. These companies further claim that their algorithms have several limitations[4] [54, 9]. For example, *Will Oremus*, a journalist at *Slate.com*, had the rare privilege of meeting the *Facebook* team in charge of the news feed[4]. He stated that the *Facebook* ranking algorithm combines hundreds of features to predict the relevance of news feed updates, but the algorithm is still likely to provide updates that users find irrelevant. Indeed, according to the journalist, following the ranking performed by the algorithm, the *Facebook* team has been running a test in which it shows hundreds of users the top update in their news feed alongside one lower-ranked update, asking them to pick the one they would prefer to read. The *Facebook* team acknowledged that the ranking performed by the algorithm corresponds "sometimes" to user preferences, declining to be more specific. When the results do not match user preferences, *Facebook* says that points to an area for improvement.

Nonetheless, if *Facebook*, *Twitter*, and *LinkedIn* are interested in ranking news feed updates, it is not only to satisfy users and retain them but also to boost the interaction rate by promoting relevant updates that are likely to make users interact[4]. Indeed, user interactions are the fuel that drives the economy of these companies as businesses are more likely to advertise their products and services on an active social platform.

3.3.2 Academic community

Ranking and predicting the relevance of news feed updates has drawn much attention over the past few years from researchers in the academic community. Indeed, we have identified no less than 15 research works [56, 58, 62, 63, 2, 64, 65, 5, 7–9, 4, 66, 67, 55, 54]. In this section, we present and discuss the most representative recent works.

In an effort to reorder and recommend relevant tweets to *Twitter* users, Shen et al. [5] and Chen et al. [64] proposed relevance prediction models that use five types of features: (1) social tie strength between the recipient user $\boldsymbol{u}$ and the author user $\boldsymbol{u'}$ (interaction rate, the similarity of social relationships, etc.); (2) the relevance of the textual content of the update $\boldsymbol{i}$ and its hashtags to the interests of $\boldsymbol{u}$; (3) the quality

of ***i*** (length, popularity, the presence of URL, etc.); (4) the authority of ***u'*** (followers and followings count, seniority, etc.); and (5) the activity of ***u'*** from the number of tweets posted. In [5], the authors first modeled the relevance of tweets by minimizing the pairwise loss of relevant and irrelevant tweets. Then, they used a supervised regression model based on a *Gradient Boosted ranking* algorithm (GBrank) [68] to predict continuous relevance scores for tweets. The model average pairwise reordering accuracy (*ACC*) was improved by 34.5% compared to the chronological model. On the other hand, in [64], the authors used a probabilistic collaborative ranking model based on *latent factors* to capture user interests and predict binary rating scores for tweets. The model's *MAP* (Mean Average Precision) was 76% and outperformed several baseline methods, including the chronological model. The results showed that recommended tweets attracted more attention than unrecommended tweets.

In order to predict the relevance of news feed updates on *Facebook*, Paek et al. [56] and Lakkaraju et al. [62] proposed prediction models that exploit three types of features: (1) social tie strength between ***u*** and ***u'***; (2) the relevance of the textual content of ***i*** to the interests of ***u***; and (3) the quality of ***i***. In [56], the authors learned binary classifier models based on supervised *Support Vector Machine* (SVM) [44] to predict the importance of news feed updates and friends. To obtain the training and evaluation data of the prediction model, the authors asked 24 users to explicitly assign relevance scores to their news feed updates. The model achieved the highest classification accuracy at 69.7%. While in [62], the authors used a collaborative filtering model based on *latent factors* to propose a scalable framework for constructing smart news feeds based on predicting user-update relevance. To obtain the training and evaluation data of the prediction model, user interactions in terms of comments were used as implicit indicators of relevance. The model achieved a precision in the range of 61.11%, indicating that the latent joint model enables the discovery of new content that is not in the immediate neighborhood.

In an attempt to rank tweets in order of relevance on *Twitter*, Uysal and Croft [63], Feng and Wang [65], De Maio et al. [54], and Vougioukas et al. [55] proposed supervised binary classifier models that use five types of features: (1) social tie strength between ***u*** and ***u'***; (2) the relevance of the textual content of ***i***, its hashtags, mentions, and URLs to the interests of ***u***; (3) the quality of ***i***; (4) the authority of ***u'***; and (5) the activity of ***u'***. First, in [63], the tweet ranking model is based on a coordinate ascent

algorithm that predicts the likelihood that a recipient user retweets a tweet from the news feed. The model average *F1 score* was 78.9% and outperformed several baseline methods. In [65], the tweet ranking model is based on collaborative filtering matrix factorization and predicts the likelihood that a recipient user retweets a tweet from the news feed. The model's *MAP* was 76.27% and outperformed several baseline methods, including the chronological model. In contrast, in [54], the relevance prediction model is based on an *artificial neural network* method attempting to re-adapt the ranking by preferring the tweets that are likely interesting to the recipient user. The model's *MAP* and *NDCG* (Normalized Discounted Cumulative Gain) outperformed several baseline methods, including the chronological model. Lastly, in [55], the prediction model is based on *logistic regression* and predicts if the recipient user may find a tweet interesting enough to retweet it. In experiments with a collection of 130K tweets received by 122 journalists, the model average *F1 score* was 90% using the *Pearson* correlation of the top ten features. In all previous work on *Twitter*, to obtain training and evaluation data, user interactions with tweets in terms of retweets and replies were used as implicit indicators of relevance.

In order to personalize news feeds on *LinkedIn*, Agarwal et al. [8] and Agarwal et al. [9] proposed supervised binary classifier models based on *logistic regression* to predict the probability that a user clicks on a post from the news feed. In [8], the proposed model uses three types of features: (1) social tie strength between $\boldsymbol{u}$ and $\boldsymbol{u'}$; (2) the quality of $\boldsymbol{i}$; and (3) the diversity of the news feed by promoting diverse updates and diverse author users. On the other hand, in [9], the proposed model uses three types of features: (1) social tie strength between $\boldsymbol{u}$ and $\boldsymbol{u'}$; (2) the relevance of the textual content of $\boldsymbol{i}$ to the interests of $\boldsymbol{u}$; and (3) the interaction rate of $\boldsymbol{u}$ with updates similar to $\boldsymbol{i}$ posted by $\boldsymbol{u'}$, which provides a finer-grained social tie strength between $\boldsymbol{u}$ and $\boldsymbol{u'}$. In the two previous works, to obtain training and evaluation data, user interactions with updates in terms of clicks were used as implicit indicators of relevance. The evaluation results showed that the click-through rate (*CTR*) of both proposed models was improved compared to the chronological model.

In an effort to assign continuous relevance scores to news feed updates on the Chinese microblogging social media *Sina Weibo* and recommend valuable tweets that users are interested in, Kuang et al. [4] proposed a ranking model based on weighted linear combinations with static feature weights. The model uses three types of features:

(1) social tie strength between ***u*** and ***u'***; (2) the relevance of the textual content of ***i*** to the interests of ***u***; and (3) the quality of ***i***. To evaluate their approach, the authors asked 1048 volunteers who are relatively active on *Sina Weibo* to explicitly assign boolean values to news feed updates (True for relevant and False for irrelevant). The model's *MAP* was 75%, outperformed several baseline methods, and was improved by 57% as compared to the results of the chronological model.

In order to predict relevance scores and rank news feed updates on the social network *SocialBlue* in [58], and an online health community in [2], the authors proposed prediction models based on weighted linear combinations with static feature weights. The proposed models exploit two types of features: (1) social tie strength between ***u*** and ***u'***; and (2) the relevance of the textual content of ***i*** to the interests of ***u***. In [58], the authors noted strong preferences for feed items presented at the top of the result lists shown to users, with 50% of selections being made on feeds in the top two positions in the feed lists and 75% of selections within the first five result positions. In [2], the average accuracy was improved by 10% compared to the results of the chronological model. In the two previous works, to obtain the evaluation data of the prediction model, user interactions in terms of authentication, comments, messages, etc., were used as implicit indicators of relevance.

In the next section, we present an in-depth analysis and synthesis of the previous research work according to four important criteria.

3.4 Analysis

According to the questions we asked in the introduction, our analysis of existing work in the academic community revolves around four criteria: (1) the features that may influence relevance; (2) the relevance prediction models; (3) the training and evaluation methods; and (4) the evaluation platforms. Table 3.1 summarizes the analysis such that the cells with the symbol "✓" indicate that the corresponding criterion was included in the research work, and the empty cells indicate that the criterion was not included. For reasons of readability, we annotate the research works in the table as follows: *'A'* for [56], *'B'* for [62], *'C'* for [58], *'D'* for [2], *'E'* for [63], *'F'* for [65], *'G'* for [5], *'H'* for [64], *'I'* for [4], *'J'* for [8], *'K'* for [9], *'L'* for [55], and *'M'* for [54]. In the rest of the section, we provide a detailed analysis according to each of the four criteria.

Work in the academic community				A	B	C	D	E	F	G	H	I	J	K	L	M
Features that may influence relevance	Social tie strength between ***u*** and ***u'***			✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	The relevance of the content of ***i***, its hashtags, mentions, and URLs to the interests of ***u***			✓	✓	✓	✓	✓	✓	✓	✓	✓		✓	✓	✓
	Quality of ***i***			✓	✓			✓	✓	✓	✓	✓	✓		✓	✓
	Authority of ***u'***							✓	✓	✓	✓				✓	✓
	Activity of ***u'***							✓		✓	✓				✓	✓
	Characteristics of ***u***								✓							
	The diversity of the news feed												✓			
	Interaction rate of ***u*** with updates similar to ***i*** posted by ***u'***													✓		
Relevance prediction models	Supervised learning models	regression with *Gradient Boosting*								✓						
		binary classification with *SVM*		✓												
		binary classification with *artificial neural networks*														✓
		binary classification with *logistic regression*											✓	✓	✓	
		binary classification with coordinate ascent algorithm						✓								
	Mathematical models	weighted linear combination				✓	✓					✓				
		Collaborative filtering models	*latent factors*		✓						✓					
			matrix factorization						✓							
Training and evaluation	Implicit				✓	✓	✓	✓	✓	✓	✓		✓	✓	✓	✓
	Explicit			✓								✓				
Evaluation platforms	*Twitter*							✓	✓	✓	✓				✓	✓
	Facebook			✓	✓											
	SocialBlue					✓										
	Online health community						✓									
	Sina Weibo											✓				
	LinkedIn												✓	✓		

Table 3.1: Research work in the academic community

3.4.1 Features that may influence relevance

We note that four types of features that may influence the relevance of the update ***i***, posted by an author ***u'***, to the recipient ***u*** were widely used in research work and approved to be effective in the prediction (see Fig. 3.3):

- Features that measure the relevance of the textual content of ***i***, its hashtags, and mentions to the interests of ***u***. Indeed, the features that match between the update content and the recipient's interests (inner product, cosine similarity, number of common words, etc.) are the most intuitive and may serve as direct predictors of relevance [4]. For example, if the user ***u*** is interested in football and the update ***i*** talks about it, then ***i*** could be relevant to ***u***.

- Features that measure social tie strength between ***u*** and ***u'***: interaction rate, number of common friends, the similarity of interests, etc. The assumption is that the update ***i*** could be relevant to the user ***u*** if ***u*** has a strong social relationship with the user ***u'*** [54]. Certainly, close friends tend to have common interests and want to catch up with the latest updates from each other [55].

- Features that measure the authority of ***u'***: followers count, followings count, listed count, seniority, etc. The assumption is that the update ***i*** could be relevant to the user ***u*** if the author ***u'*** has authority on the social media platform [65]. Indeed, Nagmoti et al. [69] state that if a user is important, i.e. has authority, then his updates are also important. For example, updates from celebrities, business leaders, journalists, and politicians are often adopted by most users [52].

- Features that measure the quality of ***i***: length, popularity, the presence of URLs, hashtags, multimedia content, etc. The assumption is that the update ***i*** could be relevant to the user ***u*** if ***i*** is of high quality (formal, informative, popular, etc.) independently of the personal interests of ***u*** [64].

3.4.2 Relevance prediction models

To predict the relevance of news feed updates, different methods and models have been used by taking the features that may influence relevance as input to produce the corresponding relevance scores as output. We note that two types of prediction models were broadly used in research work: mathematical and supervised learning models.

Mathematical models models that use mathematical tools, equations, and concepts to model and predict social behaviors [70]. We identified two main types of mathematical models: weighted linear combinations and collaborative filtering.

- In weighted linear combination models, to output a relevance score, the input features are linearly combined and summed in mathematical equations by assigning a static weight of importance to each feature according to assumptions made by the authors of the corresponding research work.
- In collaborative filtering models, we first identified *latent factor* models [26], which assume that the interactions between users and updates can be effectively captured by a low-dimensional latent representation of the user/update interaction matrix. Second, we identified matrix factorization models [27], which work by decomposing the user-update interaction matrix into the product of two lower dimensionality rectangular matrices. Indeed, *latent factors* and matrix factorization are widely used in recommender systems as a class of successful collaborative filtering models [23]. To predict relevance scores, these models attempt to find weighted low-rank approximations of the user-update interaction matrix, where weights are used to predict missing entries in the matrix. Note that the implicit and explicit relevance scores can be used in the matrix instead of interactions.

Supervised learning models machine learning algorithms that infer a function that maps an input social update to an output relevance score based on labeled training data, consisting of a set of training examples [30]. In other words, the model is trained with a history of updates that a user has found relevant or not to learn an underlying prediction function, which is able to predict whether the user will find a given update relevant. Indeed, supervised learning algorithms have been commonly used in related work and seem appropriate to effectively rank news feed updates. We identified a regression model (predict continuous relevance scores) based on a *Gradient Boosted ranking* algorithm [68], and binary classification models (predict binary relevance scores, relevant or irrelevant updates) based on *Support Vector Machine* (*SVM*) [44], *artificial neural networks* [71], *logistic regression* [40], and a coordinate ascent algorithm [72].

Indeed, mathematical and supervised learning models are suitable to predict the relevance of news feed updates since they are part of predictive analytical techniques [73]. Predictive analytics aims to model and analyze past user behaviors to make predictive

assumptions about future outcomes [74]. However, to the best of our knowledge, no comparison has been made to determine the most efficient relevance prediction models.

3.4.3 Training and evaluation methods

In research work, we identified two methods to get training and evaluation data of relevance prediction models: an implicit method and an explicit method.

- Implicit method: by assuming that a user's interaction (comment, like, share, etc.) with an update involves its relevance to that user. This method has been widely used for its ease of use since social media platforms offer by default different tools to allow users to interact and exchange with each other. Moreover, this method does not require much effort from users who naturally tend to interact with other individuals. Indeed, according to the Maslow's hierarchy of needs [75], the third level of human needs is interpersonal and implies feelings of belongingness, which include family, friendships, social interactions, etc.

- Explicit method: by asking users to assign relevance scores to their news feed updates. This method has been little used since it has several limitations. Indeed, according to Berkovsky and Freyne [1], it is not related to user interactions as user feedback was obtained via a survey or specifically developed tools on the one hand. On the other hand, this method is binding as it asks users to assign relevance scores to a large number of updates. Hence, it is unreasonable to expect users to assign relevance scores to updates unless it is rewarded. For example, in [56], 24 *Facebook* users were financially compensated for their involvement in the experiments.

3.4.4 Evaluation platforms

We find that most research works have used *Twitter*. *Twitter* is a widely used microblogging service with more than 340 million users worldwide[7]. It allows users to communicate and share their latest updates using short messages of 280 characters called *tweets* (see Fig. 3.2). The wide use of *Twitter* in related work is justified by:

- the popularity of *Twitter* as the most used microblogging service worldwide [76];

- the large flow of tweets encountered by users in their news feed [54];

[7] www.omnicoreagency.com/twitter-statistics/

- the irrelevance of a large part of tweets [61];
- the fact that social data such as user profiles, social relationships, tweets, etc. are public by default unlike other social media platforms such as *Facebook*, *LinkedIn*, and *Instagram* [77]
- the availability of API for easy data crawling [1].

Following is the only type of social relationship between *Twitter* users such that users who follow a user $\boldsymbol{u}$ are called *followers* of $\boldsymbol{u}$, and users that $\boldsymbol{u}$ follows are called *followings* of $\boldsymbol{u}$. If the user $\boldsymbol{u}$ follows another user $\boldsymbol{u'}$, $\boldsymbol{u}$ will receive in the news feed, also named *home timeline*, the tweets posted by $\boldsymbol{u'}$ and the tweets $\boldsymbol{u'}$ retweeted and/or liked. Finally, a user can perform the following three actions to interact with a tweet displayed in the news feed:

- *Retweet*: when the user wants to share the tweet with his followers;
- *Reply*: when the user wants to answer or comment on the tweet;
- *Like*: when the user wants to save the tweet in the "Likes" section.

Despite the advantages and the strengths of the different approaches proposed in related work, we have identified some limitations and open research issues that we discuss in detail in the next section.

3.5 Limitations and open issues

Following the analysis of research work, we found some limitations and identified some open research issues, which we classify according to four criteria: (1) the features that may influence relevance; (2) the relevance prediction models; (3) the training and evaluation methods; (4) and the evaluation platforms.

3.5.1 Features that may influence relevance

As discussed in section 3.4.1, we have identified four types of features that may influence the relevance of the update $\boldsymbol{i}$, posted by an author $\boldsymbol{u'}$, to the recipient $\boldsymbol{u}$ (see Fig. 3.3): (1) features between $\boldsymbol{u}$ and $\boldsymbol{i}$ which stand for the relevance of the content; (2) features between $\boldsymbol{u}$ and $\boldsymbol{u'}$ which stand for the social tie strength; (3) features of $\boldsymbol{u'}$ which

stand for the authority; and (4) features of $\boldsymbol{i}$ which stand for the quality. However, no features have been used between $\boldsymbol{u'}$ and $\boldsymbol{i}$, which we believe can stand for expertise.

To the best of our knowledge, the features that measure the expertise of $\boldsymbol{u'}$ in the topics of $\boldsymbol{i}$ have not been used in related work when predicting the relevance of updates to recipient users. The assumption is that the update $\boldsymbol{i}$ could be relevant to the recipient user $\boldsymbol{u}$ if the author $\boldsymbol{u'}$ is an expert in the topics of $\boldsymbol{i}$. According to Wagner et al. [78], updates posted by experts, i.e. users with the relevant skill or knowledge on a given topic, are often considered credible, important, and interesting. For example, *Elon Musk*, one of the most famous heroes of the tech culture, is known for his warnings about the risks of Artificial Intelligence and his updates on this subject often attract the attention of users[8]. Indeed, unlike novice users, experts usually know what they are talking about when it comes to topics they master [79].

Although expertise has not been used when predicting the relevance of updates to recipient users, the task of finding experts in specific topics has attracted much attention from researchers as a separate field, especially with the increasing growth of social media where novice and experts users coexist. Research on this topic has made significant progress in the past few decades and various techniques have been proposed. In the rest of this section, we present the most representative research work that could be leveraged to infer expertise and predict the relevance of news feed updates.

Wagner et al. [78] first explored the usefulness of different types of user-related data to make sense of the topical expertise of *Twitter* users. The authors then used topic modeling based on user-related data to build and assess the computational expertise models of users. The user-related data include: (1) aggregation of tweets a user authored or retweeted; the assumption is that a user is likely to have expertise in topics on which he frequently expresses his opinion; (2) the user biographical information, which may contain self-reported information about the education, skills, career information, etc.; and (3) information about user lists[1]; if a user is added to a list, the list label and short description of the list will appear on the user's profile. The detailed experimental analysis demonstrated that topic annotations based on biographical information are surprisingly similar to topic annotations based on the aggregation of tweets and retweets, which indicates that users tend to tweet and retweet about topics they mention in their

[8] www.wired.co.uk/article/elon-musk-artificial-intelligence-world-war-3

biography. Xu et al. [80] proposed a learning model that infers a target user's topical expertise on *Twitter* using four types of user-related data: (1) the textual content of tweets posted by the target user; (2) the location and biography of users that the target user is following; (3) the location and biography of users who follow the target user; and (4) the textual labels and descriptions of *Twitter* lists that include the target user. The experimental analysis demonstrated that each type of user data is effective for user expertise inference, with variation in performance. Moreover, aggregating multiple types of user data always achieved better performance than a single type of user data.

Liao et al. [79] explored differences between expert and novice users in inferring the expertise of *Twitter* users. In three conditions, participants rated the level of expertise of users after seeing: (1) only the textual content of tweets; (2) only the contextual information including short biographical and user list information; and (3) both tweets and contextual information. The experimental analysis demonstrated that the judgments of experts were improved when both tweets and contextual information were used. Moreover, adding tweets seemed to make little difference or even add nuance to the expert judgment of novice users. Xu et al. [81] aimed to infer the expertise of *Twitter* users based on their tweets. The work proposed a sentiment-weighted and topic relation-regularized learning model to address this problem. The model first used the sentiment intensity of a tweet to evaluate its importance in inferring a user's expertise. The intuition is that if a user can forcefully and subjectively express his opinion on a topic, the user is likely to have a strong knowledge of that topic. Then, the relatedness between expertise topics is exploited to model the inference problem. The intuition is that if a user knows the *Java* programming language, for example, the user would be closer to know the C programming language than a user who does not know *Java*. The experimental analysis demonstrated that the tweet sentiment-based weighting scheme and the topic relation regularizer make a stand-alone contribution to the model.

In the previous research work on expert finding, we notice that the biography and updates posted by a user have been broadly used to accurately infer the user's expertise. Indeed, these two user-related data are complementary. On the one hand, self-reported biographies may indicate explicit information about expertise such as education, skills, career information, etc. On the other hand, updates authored by a user may indicate implicit information about the expertise. The assumption is that a

user is likely to have expertise in topics on which he frequently expresses his opinion. These findings can be leveraged to predict the relevance of news feed updates.

3.5.2 Relevance prediction models

First, as discussed in section 3.4.2, we noticed that different supervised learning algorithms such as *Gradient Boosting*, *Support Vector Machine*, *logistic regression*, and *artificial neural networks* have been commonly used in related work and seem suitable to rank news feed updates. However, no comparison has been made to determine the most efficient relevance prediction models. Indeed, each related work intuitively chooses one supervised learning algorithm, states that the other algorithms can be used, and points out that it is out of the scope to compare different algorithms in their work. Therefore, it would be interesting to evaluate and make a comparative study of the supervised algorithms used in related work to determine the most accurate models. Moreover, with respect to supervised learning methods, it would be worthwhile to include *decision trees* [41] and *random forest* [43] algorithms in the comparison, which are known to give good results in a wide range of problems. [34]

Second, as discussed in section 3.4.2, we have identified several research works that use weighted linear combinations with static and non-personalized feature weights for all users, where the weights were set according to assumptions made by the authors of the corresponding work. However, user preferences are different on social media [1]. Therefore, the feature weights should be set in a customized manner to rank news feed updates in a personalized way according to each user's preferences. Furthermore, concerning supervised learning methods, we observed in related work that to train a prediction model, data of all users were first merged as if there is only one user. Then, a single non-personalized model has been trained on all data for all users. The importance/weight of the features learned by non-personalized models is assumed to be the same for all users, but such assumptions may not apply to some users. Indeed, we believe that non-personalized models are useful to learn the overall interests of the majority of users, but generalize such unrealistic assumptions to all users makes it difficult to predict their individual preferences. By contrast, personalized models should be trained on updates received by a particular user such that a separate model is trained per user and then employed to provide user-specific relevance scores for each new social update. Hence, we believe that using a personalized user-dependent model could enhance and refine the news feed content provided to users.

3.5.3 Training and evaluation methods

Since most research works have used the implicit training and evaluation method, which assumes that a user's interaction with an update involves its relevance, it would be interesting to study and understand its relationship with the explicit method. For example, investigate the correlation between user interactions and their explicit relevance ratings to the news feed updates. Furthermore, in the case of no user interaction, it would be interesting to include the reading time as an implicit indicator of relevance. Indeed, *Facebook* states that the more time a user spends reading an update, the more likely the update is relevant to that user[9].

3.5.4 Evaluation platforms

Since most research works have used *Twitter* as an evaluation platform, it would be interesting to exploit other types of social media that incorporate the news feed concept but have not been addressed in related work. For example, social media platforms such as *Instagram*, *Flickr*, and *Pinterest* on which photo-sharing is the aim.

In this thesis, we will focus like most related work on *Twitter* for the facilities it offers, which we have discussed in detail in section 3.4.4. However, note that it will still be possible to use the work on other social media platforms with some adaptations. Furthermore, since the explicit training and evaluation method has several limitations, we will use like most related work the implicit method for the facilities it offers, which we have discussed in detail in section 3.4.3. Moreover, as discussed in section 3.4.2, we will use supervised learning models, which have been commonly used in related work and seem appropriate to effectively rank news feed updates.

In the contributions, in addition to the four types of features used in related work, we will investigate the contribution of expertise to rank news feed updates. However, before that, we need to answer the following questions: (1) should we use a personalized or a non-personalized prediction model given that user preferences are different on social media?; and (2) which supervised prediction model should we use given that the effectiveness of the prediction depends partly on the chosen model? In the chapter 4, to answer the first question, we will introduce a novel personalized prediction model for each user and conduct a comparative study to evaluate and

[9] www.itespresso.fr/facebook-analysera-temps-lecture-publication-98813.html

compare personalized and non-personalized models. In the chapter 5, we will analyze and compare different supervised algorithms to determine the most suitable prediction models to rank news feed updates. Finally, in the chapter 6, we will exploit the findings of the two previous questions and answer the main question related to the contribution of expertise. Hence, we will introduce an approach that leverages the author's expertise that we infer from the biography and the textual content the author has posted.

3.6 Conclusion

In this chapter, we first provided background on ranking news feed updates on social media, including defining news feeds, presenting statistics on data volume and irrelevance that confirm the need for ranking, and formalizing the ranking process. Then, we discussed work carried out in the field of ranking and predicting the relevance of news feed updates in both industrial and academic communities. Afterward, we analyzed and compared the research works and exposed their advantages and limitations according to four main criteria: (1) the features that may influence relevance; (2) the relevance prediction models; (3) the training and evaluation methods; and (4) the evaluation platforms. Finally, we identified several open research issues to which we will make contributions. Following the state-of-the-art, it appears that research in the field of ranking news feed updates is not completely achieved. Indeed, several approaches have been proposed, implemented, and evaluated. However, given the limitations we identified, efforts must still be made to improve the ranking process.

In the contributions, to better rank news feed updates, we will first introduce a personalized relevance prediction model for each user and compare it with the classical non-personalized model. Then, we will compare different supervised algorithms used in related work to determine the most effective prediction models. Finally, in addition to the features used in the literature, we will leverage the expertise of the update's author for the corresponding topics when predicting the relevance.

Part II

Contributions

Chapter 4

Personalized and Non-Personalized models for RNFU: A comparative study

4.1 Introduction

In several research approaches, ranking news feed updates in descending relevance order has been proposed to help users quickly catch up with the content they may find interesting [55]. For this matter, supervised prediction models have been commonly used to predict the relevance of updates using labeled training data [82]. These models analyze past user behaviors to predict whether they will find an update relevant or not in the future [82]. However, in related work, to train a prediction model and predict the relevance, data of all users were first merged as if there is only one user. Then, a single non-personalized model has been trained on all data for all users. Indeed, according to Vougioukas et al. [55], in non-personalized models, a single global model is typically trained on a large collection of updates received by multiple users and the interactions of all users to each update, e.g. the total number of retweets per tweet. The trained global model is then used to assign a single user-independent relevance score to each new update. By contrast, personalized models should be trained on updates received by a particular user and the interactions of the particular user, e.g. whether or not the user retweeted each tweet. Hence, a separate model should be trained per user and then employed to provide user-specific relevance scores for each new tweet or, generally, social update. We believe that non-personalized models are useful to learn the overall

interests of the majority of users (e.g., users are likely to find relevant the tweets that are similar to their own tweets), but generalize such unrealistic assumptions to all users makes it difficult to predict their individual preferences. For example, a given user might be more interested in new content that is different from his own tweets. Indeed, Paek et al. [56] noticed in their study 44 cases in which several participants had rated the same news feed post and found out that 82% of the cases differ in ratings suggesting that the relevance judgment can be subjective.

In this chapter, we first present a typical approach to rank news feed updates on *Twitter* and provide a reminder of the classical non-personalized models used in related work. Then, to predict the relevance of news feed updates given that user preferences and interests are different, we introduce a novel personalized prediction model for each user based on the *random forest* algorithm. After that, we define a fair and rigorous comparison methodology by selecting the best parameters of both personalized and non-personalized models according to the collected data. Finally, we conduct a comparative study by evaluating, analyzing, and comparing personalized and non-personalized models on a real dataset crawled in a programed way from *Twitter*. The comparison and evaluation results are presented and discussed according to six criteria: (1) the overall prediction performance of both approaches to get a global overview of the most effective model; (2) the amount of data in the training set to investigate the robustness of each model; (3) the cold-start problem, which is a common problem in recommender systems; (4) the incorporation of user preferences over time; (5) the model fine-tuning to investigate the manageability of each model; and (6) the personalization of feature importance for users.

The chapter is structured as follows: section 4.2 presents a typical approach to rank news feed updates on *Twitter*, section 4.3 provides a reminder of non-personalized prediction models, section 4.4 introduces our personalized model, section 4.5 discusses the experiments we performed to compare both models and highlight the need for personalization, and section 4.6 concludes the chapter.

4.2 Typical approach for RNFU

Let $\boldsymbol{F(u)}$ denotes tweets unread by the recipient user $\boldsymbol{u}$ that can potentially be included in the news feed. Ranking news feed updates on *Twitter* implies selecting and

displaying a subset $K(u) \in F(u)$, such that $|K(u)| \ll |F(u)|$, that corresponds to the most relevant tweets to u. The rest of this chapter focuses on the most important step of the ranking, which is predicting a relevance score to each tweet $t \in F(u)$.

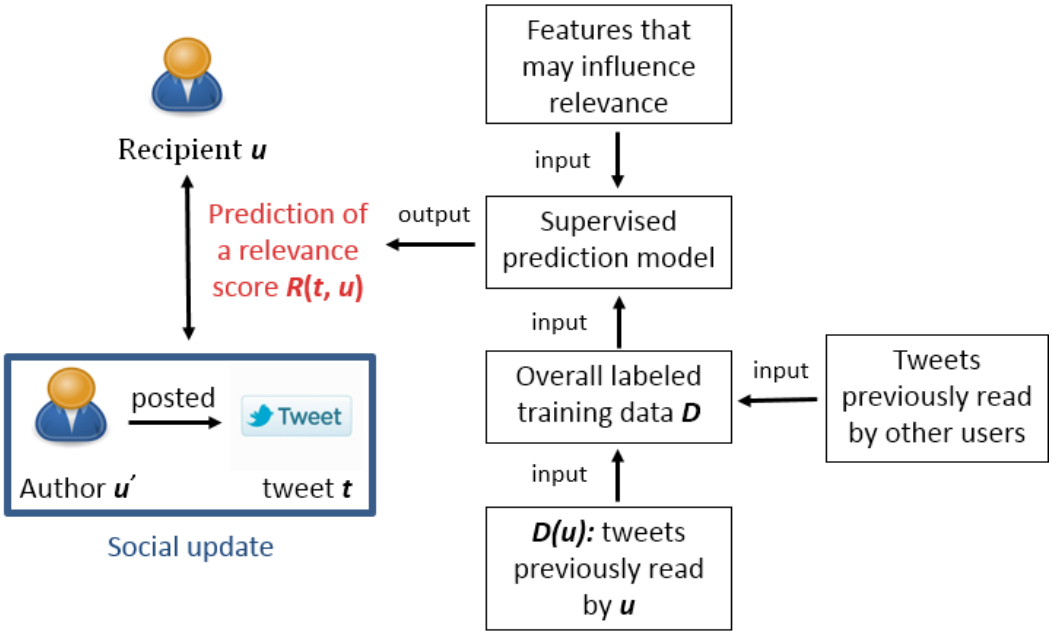

Fig. 4.1: Non-personalized prediction of a relevance score

Fig. 4.1 describes the primary non-personalized technique used in related work to predict the relevance score $R(t,u)$ of a tweet $t \in F(u)$. This technique is based on a supervised prediction model that analyzes labeled training data of tweets users read in the past to predict if a recipient user u will find the tweet t relevant in the future. Note that t is posted by an author user $u' \in A(u)$, such that $A(u)$ is the set of users that u follows. Let $D(u)$ denotes a subset of tweets previously read by the user u and D the overall labeled training data of all users. The training data of a user u is a set of input-out pairs such that an input represents a vector of features that may influence the relevance of a tweet $t' \in D$ to u, and the output represents the relevance score $R(t',u)$. The primary technique involves three steps: (1) assign relevance scores to tweets; (2) extract the features that may influence relevance; and (3) train the prediction model. In this section, we describe each step according to a typical approach that we implement and which is mainly driven by related work. Indeed, at this stage, we adopt most of the principles already used in the literature in a typical approach. The objective is first to broadly satisfy the same properties of the

approaches proposed in related work. Then, make a comparison between the classical non-personalized models and the personalized model that we introduce.

4.2.1 Relevance scores

The implicit training and evaluation method we use has been used by most related work [1]. It assumes that a previously read tweet $\boldsymbol{t' \in D(u)}$ is relevant to a user $\boldsymbol{u}$ if $\boldsymbol{u}$ interacted with $\boldsymbol{t'}$. As given by Equation 4.1, predicting relevance scores results in a binary classification problem. We believe that *likes*, as well as retweets and replies which have been used in related work, are also implicit indicators of relevance.

$$\boldsymbol{R(t',u)} = \begin{cases} 1 & \text{if } \boldsymbol{u} \text{ interacted with } \boldsymbol{t'} \text{ (retweet or reply or like)} \\ 0 & \text{otherwise} \end{cases} \tag{4.1}$$

We use the implicit method, which has been used by most related work, due to its simplicity and the fact that the explicit method used by Kuang et al. [4] has several limitations. Indeed, the explicit method is not related to user interactions on the one hand as user feedbacks were obtained via a survey or specifically developed tools. On the other hand, it is binding as it asks users to assign relevance scores to a large number of tweets. Moreover, we split relevance scores into two bins, relevant and irrelevant, because train a finer-grained classifier (e.g., $\boldsymbol{t'}$ is very relevant if $\boldsymbol{u}$ retweeted, liked, and replied to it) would be difficult as multiple user interactions with the same tweet are not frequent [55]. Indeed, out of 26180 tweets, we found that only 5% and 0% of tweets get respectively two and three types of interaction from the same user [83]. However, despite predicting binary classes, some machine learning models such as *random forest* allow to predict the probability of classes, and therefore rank tweets by relevance according to the probability of having class 1.

4.2.2 Features that may influence relevance

We extract, preprocess, and create from scratch 13 most relevant features according to related work [83]. The features may influence the relevance $\boldsymbol{R(t,u)}$ of a tweet $\boldsymbol{t}$, posted by an author $\boldsymbol{u'}$, to the recipient $\boldsymbol{u}$. The features 1, 2, 4, 5, and 7 are gradually recalculated and updated as new tweets are injected into the news feed from least recent to most recent. We thus simulate the evolution of the social media platform over time. The features are summarized in Table 4.1 and divided into four categories:

Table 4.1: Features that may influence relevance

Features that may influence relevance		**Type**	**N°**
Relevance of the content of ***t***, its hashtags, and mentions to ***u***	Relevance of the keywords of ***t*** to ***u***	Int	*f1*
	Relevance of the hashtags of ***t*** to ***u***	Int	*f2*
	Presence of ***u*** in the mentions of ***t***	Bool	*f3*
Social tie strength between ***u*** and ***u'***	Interaction rate of ***u*** with tweets of ***u'***	Float	*f4*
	Number of times ***u*** mentioned ***u'***	Int	*f5*
Authority of ***u'***	Followers count / Followings count	Int	*f6*
	Seniority in years	Int	*f7*
	Listed (group) count	Int	*f8*
Quality of ***t***	Length (# characters)	Int	*f9*
	Presence of hashtags	Bool	*f10*
	Presence of a URL	Bool	*f11*
	Presence of a photo or a video	Bool	*f12*
	Popularity (# retweets, replies, likes)	Int	*f13*

- Features that match between the content of ***t*** and the interests of ***u***.
- Features that measure social tie strength between ***u*** and ***u'***. The assumption is that ***t*** could be relevant to ***u*** if ***u*** and ***u'*** are close friends.
- Features that measure the authority of ***u'***. The assumption is that ***t*** could be relevant to ***u*** if ***u'*** has authority. Indeed, the more important a user, the important the tweets posted by that user.
- Features that measure the quality of ***t***: length, popularity, the presence of multimedia content, etc. The assumption is that ***t*** could be relevant to ***u*** if it is of high quality.

In the rest of the section, we provide a detailed description of each feature.

Relevance of the keywords of t to u According to Shen et al. [5], keywords of tweets posted by a user and/or with which he interacted implicitly reflect his topics of interest and may serve as direct predictors of relevance. First, after removing HTML characters and URLs in the textual content of the tweet $\boldsymbol{t}$, we represent its topics by keywords extracted using the *DBpedia Spotlight* annotation service[1]. This annotation service is an open-source project for extracting structured semantic entities from *Wikipedia*. More details are provided in [84]. This method has been proven not to be compromised by the short and informal nature of tweets [85], unlike methods used in related work, which are mainly based on TF-IDF [65, 55, 63] or a topic model [64, 5]. Each keyword in a tweet is represented by the URI[2] (Uniform Resource Identifier) corresponding to the annotated entity. In the tweet of Fig. 6.1 for example, using a *support* = 20 and a *confidence* = 0.32, the annotation service outputs the following keywords: *China, Russia, Computer_Science, Artificial_Intelligence, World_War_III*. Finally, we propose to calculate this feature as follows:

$$f_1(\boldsymbol{u},\boldsymbol{t}) = \sum_{i=1}^{nbk(\boldsymbol{t})} P(\boldsymbol{u}, k_i(\boldsymbol{t})) \tag{4.2}$$

Where:

- $k_i(\boldsymbol{t})$ is the i^{th} keyword of $\boldsymbol{t}$
- $nbk(\boldsymbol{t})$ is the number of keywords of $\boldsymbol{t}$
- $P(\boldsymbol{u},\ k_i(\boldsymbol{t}))$ is the number of times $\boldsymbol{u}$ has previously posted and/or interacted (retweet, reply, like) with $k_i(\boldsymbol{t})$

For example, if $\boldsymbol{t}$ has 2 keywords k_1 and k_2, and $\boldsymbol{u}$ has previously posted and/or interacted 10 times with k_1 and 5 times with k_2, then the sum of the two values, i.e. 15, will be assigned to this feature.

Relevance of the hashtags of t to u According to Chen et al. [64], hashtags of tweets posted by a user and/or with which he interacted implicitly reflect his topics of interest and may serve as key predictors of relevance. To calculate this feature, we first lowercase hashtags and then propose the following equation:

[1] http://demo.dbpedia-spotlight.org/
[2] A string of characters used to identify a resource.

$$f_2(\boldsymbol{u},\boldsymbol{t}) = \sum_{i=1}^{nbh(\boldsymbol{t})} P(\boldsymbol{u}, h_i(\boldsymbol{t})) \tag{4.3}$$

Where:

- $h_i(\boldsymbol{t})$ is the i^{th} hashtag of $\boldsymbol{t}$
- $nbh(\boldsymbol{t})$ is the number of hashtags of $\boldsymbol{t}$
- $P(\boldsymbol{u},\ h_i(\boldsymbol{t}))$ is the number of times $\boldsymbol{u}$ has previously posted and/or interacted (retweet, reply, like) with $h_i(\boldsymbol{t})$

For example, if $\boldsymbol{t}$ has 2 hashtags h_1 and h_2, and $\boldsymbol{u}$ has previously posted and/or interacted 5 times with h_1 and 3 times with h_2, then the sum of the two values, i.e. 8, will be assigned to this feature.

Presence of $\boldsymbol{u}$ in the mentions of $\boldsymbol{t}$ A tweet that mentions the user $\boldsymbol{u}$ is likely to match his interests and be relevant. Feng and Wang [65] state that a mention serves to draw a user's attention to the corresponding tweet. The authors propose to calculate this feature with a boolean variable.

Interaction rate of $\boldsymbol{u}$ with tweets of $\boldsymbol{u'}$ According to Vougioukas et al. [55], if $\boldsymbol{u}$ interacted (retweet, reply, like) frequently with tweets posted by $\boldsymbol{u'}$ in the past, i.e. $\boldsymbol{u}$ found the tweets of $\boldsymbol{u'}$ relevant, then they tend to have a strong social relationship, and thus $\boldsymbol{u}$ may find the tweets of $\boldsymbol{u'}$ relevant in the future. We propose to calculate this feature with the following equation:

$$f_4(\boldsymbol{u},\boldsymbol{u'}) = \frac{|\text{Tweets posted by } \boldsymbol{u'} \text{ with which } \boldsymbol{u} \text{ interacted}|}{|\text{Tweets posted by } \boldsymbol{u'} \text{ that } \boldsymbol{u} \text{ has previously read}|} \tag{4.4}$$

Number of times $\boldsymbol{u}$ mentioned $\boldsymbol{u'}$ According to Chen et al. [64], if the user $\boldsymbol{u}$ mentioned the user $\boldsymbol{u'}$ in the past, then $\boldsymbol{u}$ pays attention to $\boldsymbol{u'}$, the two users may thus have a strong social relationship. The user $\boldsymbol{u}$ may therefore find the tweets of $\boldsymbol{u'}$ relevant in the future. Chen et al. [64] propose to calculate this feature by counting the number of times $\boldsymbol{u}$ mentioned $\boldsymbol{u'}$ in tweets he posted.

Followers count / Followings count Pan et al. [52] assert that users who have authority such as celebrities, business leaders, journalists, and politicians tend to

have more followers than followings. The authors propose to calculate this feature by dividing the followers count of ***u'*** by the followings count.

Seniority Shen et al. [5] state that senior users, i.e. users whose accounts were created early on social media, tend to have more authority than new incoming users. We propose to calculate this feature using Equation 4.5. The assumption is the longer a user has been on the social media, the more experienced and authoritative the user is, and the more relevant the tweets posted by that user.

$$f_7(\boldsymbol{u'},\boldsymbol{t}) = \text{Year in which } \boldsymbol{t} \text{ was posted} - \text{Year in which the account of } \boldsymbol{u'} \text{ was created} \tag{4.5}$$

Listed count *Twitter* lists allow users to organize people they follow into labeled groups [86]. According to Duan et al. [87], the number of lists to which a user has been added is an adequate representation of the authority on social media. Uysal and Croft [63] propose to assign the list count of ***u'*** to this feature.

Length Chen et al. [64] assert that a long tweet is likely to be more formal, more informative, and of better quality than a short tweet. Vougioukas et al. [55] propose to compute the length of the tweet ***t*** by its number of characters.

Presence of hashtags A tweet with hashtags can provide more information and be of better quality. Indeed, hashtags help group tweets and conversations around a similar topic so users can easily find and follow what interests them. Chen et al. [64] state that the author spent time tagging the tweet thinking it might be useful. We propose to calculate this feature with a boolean variable.

Presence of a URL De Maio et al. [54] state that since tweets are limited to 280 characters, users tend to include a URL to a website containing more details. According to the authors, a tweet with a URL can give more information and be of better quality. They propose to calculate this feature with a boolean variable.

Presence of a photo or a video Feng and Wang [65] assert that a tweet with multimedia content can give more details and be of good quality. The authors propose to calculate this feature with a boolean variable.

Popularity Kuang et al. [4] state that the more users interact with a tweet, the better its quality. We propose to calculate this feature as follows:

$$f_{13}(\boldsymbol{t}) = \text{Number of retweets of } \boldsymbol{t} + \text{Number of replies to } \boldsymbol{t} + \text{Number of likes of } \boldsymbol{t} \tag{4.6}$$

4.2.3 Relevance prediction model

The prediction model aims to analyze labeled training data of tweets users read in the past to predict if they will find a tweet relevant in the future. Let $\boldsymbol{S}$ denotes the set of recipient users. First, we generate training data instances for each recipient user $\boldsymbol{u} \in \boldsymbol{S}$ in the form of input-output pairs considering each previously read tweet $\boldsymbol{t'} \in \boldsymbol{D(u)}$. An input represents a vector of features that may influence the relevance of $\boldsymbol{t'}$ to $\boldsymbol{u}$, and the output represents the implicit relevance score $\boldsymbol{R(t',u)}$. Then, we can either train a personalized prediction model for each user $\boldsymbol{u} \in \boldsymbol{S}$, or merge all data as if there was only one user to train a single non-personalized model for all users. The aim of both approaches is to map new input features of a tweet unread by a user $\boldsymbol{u}$ to a relevance score using a binary classifier learned from previously read tweets in the training set. In the next section, we provide a reminder of non-personalized models.

4.3 Non-personalized models

According to Vougioukas et al. [55], in non-personalized models, a single global model is typically trained on a large collection of tweets received by multiple users and the interactions of all users to each tweet. The trained global model is then used to assign a single user-independent relevance score to each new tweet. Fig. 4.2 describes the primary technique used in related work to train a non-personalized prediction model. First, historical user data, which consists of previously read tweets $\boldsymbol{D}_i$, are merged and scaled to have feature values within the same range. Then, the overall data $\boldsymbol{D}$ is shuffled as if there were only one user and no chronological order of tweets. Finally, data is split into two sets to train the prediction model: a training set for 70% of the data and a test set for 30% of the remaining data to evaluate the performance.

Table 4.2 indicates the non-personalized models used in related work in contrast to the personalized model we propose. The table shows that different supervised algorithms were used: *logistic regression* [8, 9, 88, 55], *Support Vector Machines* [56],

Table 4.2: Non-personalized models in related work vs. Personalized model

Research work	Data	Supervised algorithm	A prediction model for
[63]	665 tweets	Coordinate ascent algorithm	All users
[5]	816 users	*Gradient Boosting*	
[7]	675 users	*Naive Bayes*	
[55]	122 users	*logistic regression*	
[54]	2 users	*artificial neural networks*	
[67]	307 users		
[66]	1000 users		
[56]	24 users	*Support Vector Machines*	Each fold of data (5 folds)
[8]	LinkedIn users	*logistic regression*	Each partition of data (3 partitions)
[9]			
Facebook [88]	Trillions of examples	*logistic regression, artificial neural networks, Gradient Boosting, etc.*	Each demographic subset of users
Personalized model	46 users	*Random Forest*	Each user

artificial neural networks [88, 66, 67, 54], etc. In each work, a single algorithm was used for either: all users [63, 5, 7, 66, 55, 67, 54], each fold/partition of data with five folds [56] and three partitions [8], or each demographic subset of users [88]. In other words, no related work has used a single model for each user, such that in the best of cases, five models were used for all users in [56] and n models in [88], where n is the number of demographic subsets of users. Furthermore, the models used in related work use different sizes of data from 2 [54] to 1000 users [66]. Note that *Facebook* and *LinkedIn* data science teams are an exception as they have access to millions of users and trillions of examples [8, 9, 88]. The research work state that non-personalized models benefit from a large collection of tweets in the training set. Each tweet is represented as a feature vector that includes user-specific features. If two users receive the same tweet, it will be represented by two different feature vectors, which allows the model to produce different predictions per user for the same incoming tweet.

Nonetheless, since non-personalized models are trained on all data as if there is only one user, the models may learn and generalize unrealistic assumptions (e.g., all users are likely to find relevant the tweets that are similar to their own tweets). The importance/weight of the features learned by non-personalized models is assumed to be the same for all users, but such assumptions may not apply to some users. For

example, a given user might be more interested in new content that is different from his own. Indeed, Paek et al. [56] asked 24 participants to rate news feed posts and noticed that 82% of ratings that concern the same tweets are different. This study indicates that the relevance judgment is subjective as user preferences and interests are different. Therefore, we believe that using a personalized user-dependent model could be crucial to enhance and refine the news feed content. In the next section, we use *random forest* to train and introduce a personalized prediction model for each user.

4.4 A personalized prediction model

In contrast to non-personalized models, personalized models should be trained on tweets received by a particular user and the interactions of the particular user to each tweet. Hence, a separate model should be trained per user and then employed to provide user-specific relevance scores for each new tweet. Fig. 4.2 describes the technique we use to train a personalized prediction model for each user and assign user-specific relevance scores to tweets. First, we sort tweets by time and divide the training data $\boldsymbol{D}_i$ of each user $\boldsymbol{u}_i \in \boldsymbol{S}$ into two sets: a training set of the prediction model for the 70% least recent instances and a test set for the 30% remaining most recent instances. The purpose is to keep a chronological track of the relevance judgment of tweets by users over time. Then, we use the training set of each user $\boldsymbol{u}_i \in \boldsymbol{S}$ to train the corresponding *random forest* model $\boldsymbol{M}_i$. *Random Forest* [43] is a popular *ensemble learning* method[3] for classification and regression problems that operate by constructing a multitude of *decision trees*. We choose *random forests* as prediction models because they [89]: (1) tend to result in powerful prediction models, especially for binary classification problems; (2) do not need data preprocessing; (3) do not require parameterization; (4) are fast to train; (5) can handle a large number of features; (6) implicitly perform feature selection; (7) seldom overfit; and (8) allow to compute feature importance in judging the relevance of tweets by users. Note that at this stage, other supervised learning algorithms are also applicable and that it is out of the scope of this chapter to compare them.

The aim of using a personalized *random forest* model for each user is to make tailored recommendations, which may not coincide with the interests of the majority of users that non-personalized models are trained to predict. Indeed, unlike non-

[3] A method that uses multiple machine learning algorithms to obtain better predictive performance than could be obtained from any of the constituent learning algorithms alone.

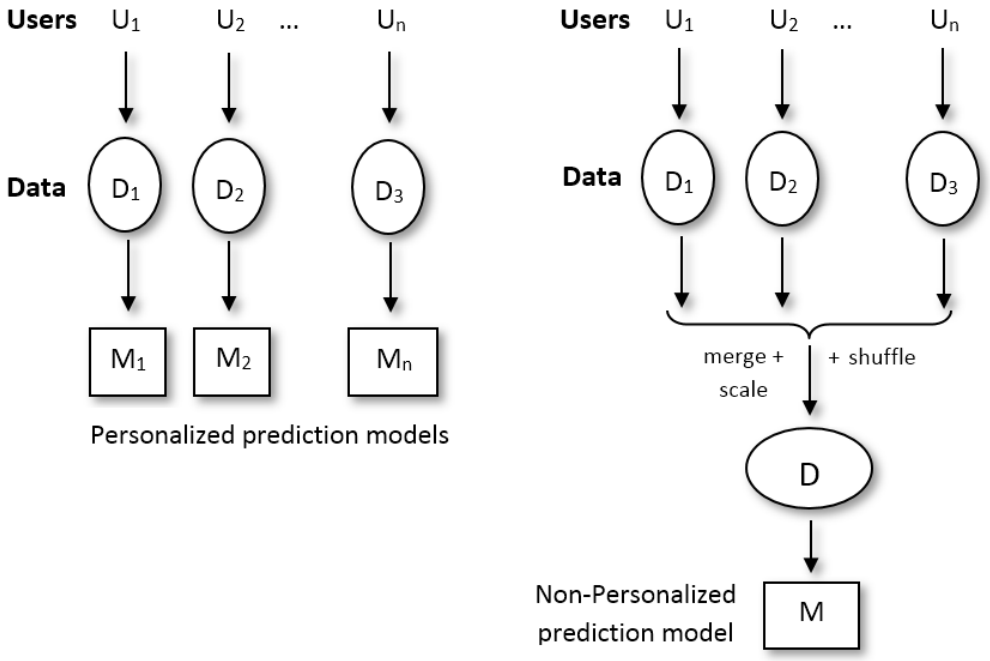

Fig. 4.2: Personalized and Non-Personalized models

personalized models, not only the feature vector is different for each user-tweet pair, but also the feature importance/weight for each user. In other words, as a model is trained on the data of a given user independently of the other users, the model learns the individual user preferences and interests (e.g., a user interested in art is more likely to find tweets with multimedia content relevant). Another reason to use a personalized model for each user is to sort and split the corresponding train and test data by time. Train the model on recent data allows to track changes in user preferences over time and make time-sensitive recommendations accordingly. In the next section, we describe the experiments we used to compare personalized and non-personalized models.

4.5 Experiments and comparison results

To compare personalized and non-personalized models and highlight the need for personalization, we describe in this section: (1) the *Twitter* dataset we used in the experiments; (2) the measures we used to evaluate the performance; (3) the methodology we used in the comparison; and (4) the obtained results.

4.5.1 Dataset

To collect real data on *Twitter*, we first randomly selected a set $\boldsymbol{S}$ of 46 *Twitter* users so that each user met the following criteria:

- has a great number of social relationships, at least 20 followings, which we believe is the minimum number. Ranking news feed updates is proposed to help these kinds of users catch up with the relevant updates. Note that the average number of followings is 72 followings per user;
- interacts (retweet, reply, like) frequently with tweets from the news feed with an interaction rate greater than 10%, which we believe is the minimum rate. This criterion is due to the use of the implicit training and evaluation method, which assumes that user interaction with a tweet involves its relevance;
- English-speaking in order to use the English version of *DBpedia Spotlight*, which is the most efficient [90];
- the tweets the user posts are public and visible to all users, unlike protected tweets that are only visible to the followers.

Then, using *Twitter Rest API*[4], we collected over ten months all data needed for the proposed approach: (1) the explicit profile of each recipient user $\boldsymbol{u} \in \boldsymbol{S}$, including registration date, followers, followings, and listed count; (2) the explicit profile of each user $\boldsymbol{u'} \in \boldsymbol{A(u)}$, such that $\boldsymbol{A(u)}$ is the set of users that $\boldsymbol{u}$ follows; and (3) the tweets posted by each user $\boldsymbol{u} \in \boldsymbol{S}$ and the tweets posted by his followings. Finally, the tweet keywords were extracted using the English version of *DBpedia Spotlight* with a *support* = 20 and a *confidence* = 0.32 (values determined through experiments).

Nonetheless, it is impossible to directly retrieve the news feed of a particular user on *Twitter* and tell in case of non-interaction if the user has read a given tweet. Therefore, to simulate the news feed of each recipient user $\boldsymbol{u} \in \boldsymbol{S}$, we used a variant of the principle proposed by Feng and Wang [65] to select, $\boldsymbol{D(u)}$, the subset of tweets posted by the followings of $\boldsymbol{u}$ that $\boldsymbol{u}$ may have read. The tweets read by $\boldsymbol{u}$ with which he did not interact are considered irrelevant according to the implicit training and evaluation method. Note that we did not select the tweets retweeted and/or liked by

[4]https://dev.twitter.com/rest/public

the followings of $\boldsymbol{u}$ since the interactions of $\boldsymbol{u}$ with these tweets are redirected to users who do not necessarily belong to the social network of $\boldsymbol{u}$[5]. The variant is as follows:

- First, sort all the tweets posted by the followings of $\boldsymbol{u}$ in chronological order from least recent to most recent;
- Then, for each tweet $\boldsymbol{t'}$ with which $\boldsymbol{u}$ interacted using a retweet, reply, or like, keep the chronological session defined by the tweet $\boldsymbol{t'}$, the tweet before $\boldsymbol{t'}$, and the tweet after $\boldsymbol{t'}$;
- Finally, after deleting duplicates, sort the selected tweets again in chronological order from least recent to most recent.

The use of our variant resulted overall in 26180 tweets. Furthermore, the variant resulted in about 35% interaction rate with tweets, with an average of 569 tweets and a median of 343 tweets as training data instances for each recipient user.

4.5.2 Measures

First, as described in sections 4.3 and 4.4, we train *random forest* classifiers for both personalized and non-personalized models using the corresponding training set with 70% of the data. Then, to evaluate the models using the corresponding test set with 30% of the data, we define the confusion matrix of Table 4.3 and present below the related concepts [91]:

Relevance (R)		**Predicted class**	
		$R = 0$	$R = 1$
Actual class	$R = 0$	True Negative (TN)	False Positive (FP)
	$R = 1$	False Negative (FN)	True Positive (TP)

Table 4.3: Confusion matrix

- True Positive (TP): # of relevant tweets correctly predicted relevant
- True Negative (TN): # of irrelevant tweets correctly predicted irrelevant

[5]If a user *Karim* retweet a tweet posted by *Mohamed*, and *Mehdi* retweet the retweet of *Karim*, the retweet of *Mehdi* will be redirected to the tweet of *Mohamed*.

- False Positive (FP): # of irrelevant tweets incorrectly predicted relevant
- False Negative (FN): # of relevant tweets incorrectly predicted irrelevant

After that, we use the weighted *F1 score* given by Equation 4.7 [91], which is a popular measure for binary classification. The equation calculates the standard *F1 score* for each class and finds their average weighted by support, i.e. the number of true instances for each class. This metric is suitable to evaluate the performance since classes are slightly unbalanced with an interaction rate with tweets of about 35% for each user. Moreover, we are interested in measuring the performance of predicting both relevant and irrelevant tweet classes.

$$F = \frac{(F_r \times (TP + FN)) + (F_i \times (TN + FP))}{TP + TN + FP + FN} \tag{4.7}$$

Where:

- F_r is the standard *F1 score* for the class of relevant tweets
- F_i is the standard *F1 score* for the class of irrelevant tweets

4.5.3 Methodology

In the experiments, we first selected the best *random forest* parameters (number of trees, maximum three depth, splitting criterion, etc.) for a fair comparison between non-personalized and personalized models. Indeed, a *random search* was run over different parameter values so that the parameters are optimized by a cross-validated search over parameter settings. *Random search* is a widely used strategy for algorithm hyper-parameter optimization that randomly samples values from the statistical distribution of each parameter to select the values that give the best performance according to the cross-validation [92]. In the K-fold cross-validation, the dataset is shuffled by default and then split into K different folds. Each fold is then used once as a test set while the K - *1* remaining folds form the training set [34] (see Fig. 4.3). The K evaluation results can then be averaged to produce a single estimation. Therefore, to select the best parameters, we used 5-fold cross-validation for the non-personalized model and 5-fold time-series cross-validation for the personalized model, both performed on the train set [91]. As shown in Fig. 4.3, a time-series validation was used for the personalized model because it preserves the chronological order of tweets, unlike the non-personalized model where data is shuffled. Indeed, in the time-series validation, the least recent

data form the train set, and the most recent data are used as a test set.

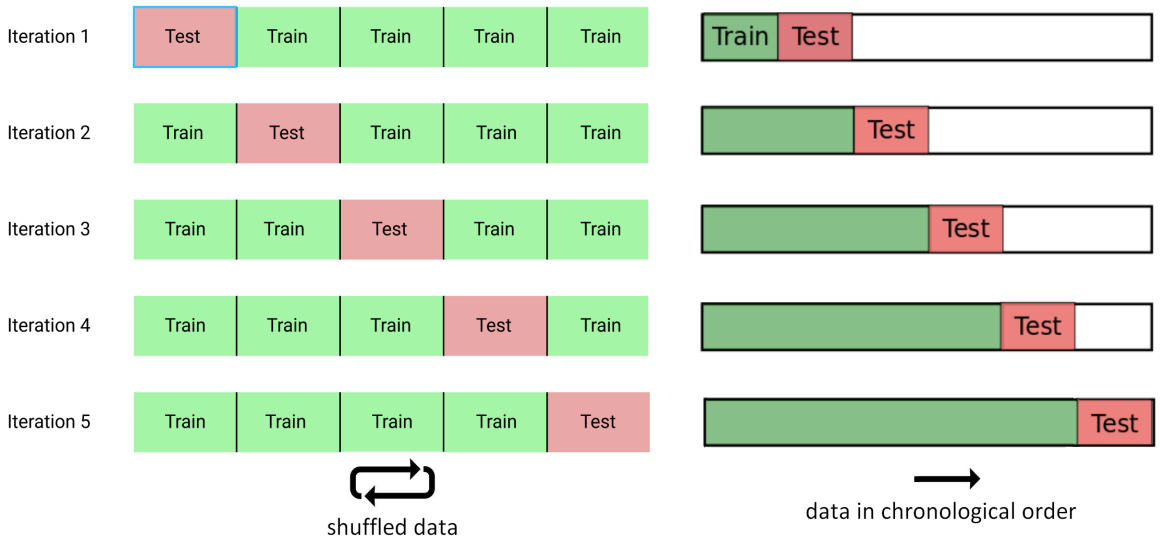

Fig. 4.3: Cross-validation (left) and Time-series cross-validation (right)

Then, to study the algorithmic stability with several runs and small changes to training data, we retrained and iterated the best models on 30 different *random state*[6] values and evaluated them on the test set. The average *F score* was selected for both personalized and non-personalized approaches.

4.5.4 Results

The comparison and evaluation results are presented and discussed according to six criteria: (1) the overall prediction performance of both approaches to get a global overview of the most effective model; (2) the amount of data in the training set to investigate the robustness of each model; (3) the cold-start problem, which is a common problem in recommender systems; (4) the incorporation of user preferences over time; (5) the model fine-tuning to investigate the manageability of each model; and (6) the personalization of feature importance for users.

First, Table 4.4 summarizes the pros and cons of non-personalized and personalized models according to all evaluation results. The results show that introducing a

[6] A variable used in randomized machine learning algorithms to determine the random seed of the pseudo-random number generator.

Table 4.4: Personalized vs. Non-Personalized models

	Non-personalized model	Personalized models
PROS	- More data $\Longrightarrow$ A more robust model - Address the problem of cold-start and inactive users - Easy to fine-tune a single model	- Higher *F score* than non-personalized (80.85) - Time-aware user preferences - Different feature importance for each user
CONS	- Lower *F score* than personalized (77.73) - Time-unaware user preferences - The same feature importance for all users	- Less data $\Longrightarrow$ Less robust models - The problem of cold-start and inactive users - Difficult to fine-tune several models

personalized user-dependent model has improved the average *F score* by +3.12%, from an *F score* of 77.73% with the non-personalized model to an *F score* of 80.85% with the personalized model. Therefore, to make refined predictions and select the tweets that might be relevant to a given user, it is more convenient to train a model on tweets the user has found relevant in the past rather than including in the training process tweets and behaviors about other users. Undoubtedly, tweets that are relevant to one user are not necessarily relevant to another user, which illustrates the importance of the personalized model we introduce to capture individual user needs and improve the prediction accuracy. Time-aware user preferences are another advantage of personalized models that makes them more accurate. Indeed, train the model on recent data allows time-sensitive recommendations. The personalized models capture the chronological evolution of user relevance judgment of tweets, which may change with time (e.g., a user may over time give less importance to popular tweets and more importance to tweets related to his interests). In contrast, the non-personalized model used in literature does not predict such behaviors since data of all users are merged and shuffled as if there were only one user and no chronological order of tweets.

Second, we computed feature importance values[7] [43] in both personalized and non-personalized models, which are presented in Table 4.5 and Fig. 4.4 respectively. Fig. 4.4 gives the average feature importance for all recipient users. As shown in the figure, non-personalized models learn and provide an overview of the features that influence the relevance judgment of tweets by users, which is useful to understand user behaviors and the assessment of relevance in general. For example, the results show that the top feature is the feature f_4 (0.5), the interaction rate of $\boldsymbol{u}$ with tweets posted

[7] *Random Forest* computes the importance of a feature as the normalized total reduction of the criterion brought by that feature, also known as the *Gini importance*.

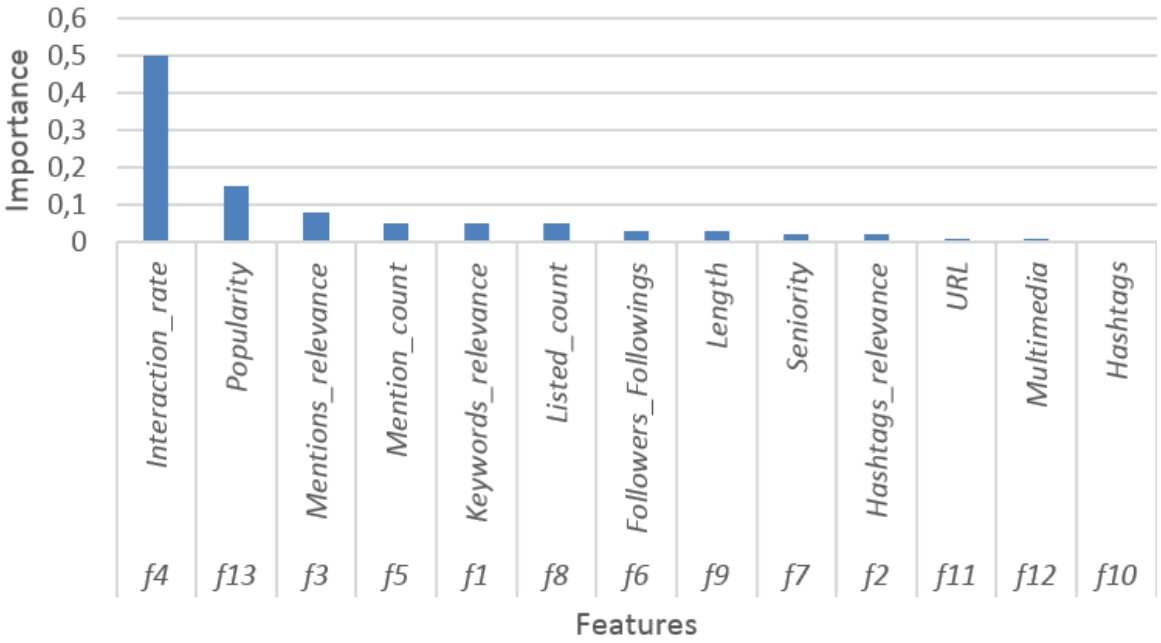

Fig. 4.4: Non-Personalized feature importance

by ***u'***. Certainly, if ***u*** found tweets posted by ***u'*** relevant in the past, then ***u*** may find the tweets of ***u'*** relevant in the future. The second most important feature is the feature f_{13} (0.15) which measures the popularity of the tweet. Indeed, the more users interact with a tweet, the more likely it is to be of high quality. The third most important feature is the feature f_3 (0.08) which tells whether ***u*** is mentioned in the tweet ***t***. Feng and Wang [65] state that a mention draws a user's attention to a tweet that is likely to match his interests. The results also reveal that the features f_5 (0.05) which represents the number of times ***u*** mentioned ***u'*** in tweets he posted is important. Indeed, if ***u*** mentioned ***u'*** in the past, then they tend to have a strong social relationship, and ***u*** may find the tweets of ***u'*** relevant in the future. The results also indicate that the features f_6, f_7, f_8, and f_{12} (0.03, 0.05, and 0.02 respectively) which measure the author's authority are important. These features are consistent with the observations in [69] that state that if a user is important, then his updates could also be important. We further note that the features f_1 and f_2 (0.05 and 0.02 respectively), which measure respectively the relevance of the content of ***t*** and its hashtags to ***u***, are surprisingly not the most important. This proves that predicting relevance scores is a difficult task because the most important features are not the most intuitive. Finally, we notice that the features f_9, f_{10}, f_{11}, and f_{12} (0.03, 0, 0.01, and 0.01 respectively), which measure tweet quality, are not important since they are

not related to users and do not take into consideration their preferences [65].

In contrast to non-personalized feature importance, Table 4.5 gives the personalized feature importance for each recipient user. First, we note that feature importance differs according to users, i.e. features that are important to one user are not necessarily important to another user, e.g. the feature f_9 which stands for the tweet length is very important to the user *Red_or_MC1R* when judging the relevance of tweets (0.22) but not to the user *Medium* (0.02). Certainly, user preferences are different, and this illustrates the gain brought by a personalized prediction model for each user, which takes into consideration individual interests. Furthermore, we note that the features learned as highly important by the non-personalized model are in fact not important to all users. For example, the top feature f_4 (0.5), the interaction rate of $\boldsymbol{u}$ with tweets posted by $\boldsymbol{u'}$, is important to many users when judging the relevance of tweets, indeed, but not to some users, e.g. the users *TheMuslimReform* (0.01), *LKrauss1* (0.02), and *bamwxcom* (0.03). This proves that non-personalized models generalize unrealistic assumptions to all users. In opposite, the personalized models we introduce allow tailored recommendations that do not coincide with the preferences of the majority of users that non-personalized models are trained to predict.

Despite all the improvements the personalized models have brought in, we observe from the evaluation results that the proposed approach has some limitations. Fig. 4.5 presents the learning curve[8] of the non-personalized model for all users and the learning curve of the personalized model for the user *ch402*. The learning curves of the 46 users in the dataset are quite similar; hence we selected one user as a case study.

First, Fig. 4.5 shows that the non-personalized model benefits from a large collection of tweets in the training set compared to the personalized model (20000 against 400 tweets). Indeed, unlike personalized models which are trained on the individual data of each user, the non-personalized model merges the data of all users, which allows it to be trained on a large collection of tweets received by different users. Note that in the dataset, there is an average of 569 tweets in the training database of each recipient user and a median of 343 tweets.

[8] A learning curve shows the validation and training score of an estimator for varying numbers of training samples.

Table 4.5: Personalized feature importance

User	*f1*	*f2*	*f3*	*f4*	*f5*	*f6*	*f7*	*f8*	*f9*	*f10*	*f11*	*f12*	*f13*
Astro0Glen	0.02	0.04	0.0	0.39	0.06	0.13	0.07	0.1	0.04	0.01	0.01	0.0	0.14
Astro_Pam	0.05	0.06	0.03	0.23	0.06	0.09	0.05	0.1	0.09	0.01	0.03	0.02	0.2
bamwxcom	0.0	0.01	0.13	0.03	0.0	0.1	0.06	0.08	0.08	0.01	0.02	0.08	0.39
Baronatrix	0.03	0.01	0.0	0.21	0.0	0.11	0.09	0.15	0.12	0.01	0.04	0.01	0.23
BethStamper6	0.04	0.0	0.0	0.3	0.0	0.06	0.06	0.1	0.14	0.02	0.04	0.02	0.22
byudkowsky	0.03	0.0	0.0	0.14	0.12	0.21	0.05	0.12	0.17	0.0	0.01	0.0	0.14
ch402	0.03	0.0	0.01	0.19	0.02	0.11	0.07	0.15	0.13	0.01	0.02	0.01	0.25
demishassabis	0.06	0.0	0.02	0.25	0.0	0.09	0.23	0.09	0.08	0.01	0.01	0.0	0.17
eevil_abby	0.2	0.0	0.07	0.06	0.05	0.06	0.05	0.09	0.19	0.02	0.01	0.03	0.18
elonmusk	0.04	0.0	0.0	0.23	0.02	0.1	0.06	0.07	0.06	0.01	0.02	0.01	0.39
GeorgeHarrison	0.0	0.01	0.0	0.2	0.22	0.07	0.03	0.15	0.02	0.0	0.0	0.0	0.29
GilmoreGuysShow	0.03	0.0	0.0	0.1	0.0	0.14	0.08	0.19	0.13	0.02	0.07	0.01	0.22
gwern	0.03	0.0	0.0	0.21	0.01	0.16	0.07	0.11	0.08	0.04	0.01	0.0	0.29
homebrew	0.02	0.01	0.01	0.2	0.09	0.07	0.03	0.1	0.08	0.01	0.01	0.01	0.37
HybridZizi	0.04	0.0	0.0	0.3	0.0	0.11	0.06	0.13	0.1	0.02	0.03	0.02	0.21
jadelgador	0.08	0.01	0.0	0.29	0.04	0.07	0.05	0.09	0.03	0.01	0.0	0.01	0.33
JHUBME	0.05	0.04	0.11	0.17	0.13	0.08	0.04	0.11	0.07	0.01	0.0	0.0	0.19
JohnDawsonFox26	0.04	0.02	0.0	0.26	0.12	0.07	0.04	0.06	0.05	0.01	0.02	0.08	0.25
john_walsh	0.0	0.21	0.02	0.13	0.03	0.14	0.03	0.17	0.05	0.03	0.0	0.0	0.18
kilcherfrontier	0.01	0.12	0.12	0.27	0.07	0.07	0.04	0.19	0.03	0.0	0.02	0.0	0.06
LKrauss1	0.05	0.0	0.08	0.02	0.06	0.11	0.06	0.15	0.17	0.04	0.14	0.01	0.12
mastenspace	0.14	0.0	0.3	0.23	0.01	0.05	0.02	0.17	0.02	0.0	0.0	0.0	0.05
Medium	0.4	0.0	0.21	0.06	0.0	0.03	0.01	0.16	0.02	0.02	0.0	0.0	0.09
microphilosophy	0.13	0.01	0.0	0.11	0.01	0.12	0.16	0.08	0.1	0.02	0.01	0.01	0.25
MIRIBerkeley	0.23	0.0	0.0	0.09	0.04	0.09	0.18	0.12	0.08	0.01	0.01	0.0	0.15
NASAKepler	0.12	0.07	0.31	0.03	0.06	0.04	0.01	0.15	0.1	0.0	0.0	0.0	0.1
NASA_Wallops	0.08	0.05	0.03	0.11	0.02	0.07	0.0	0.18	0.03	0.0	0.05	0.0	0.38
newscientist	0.26	0.0	0.19	0.07	0.0	0.08	0.03	0.1	0.07	0.01	0.0	0.02	0.18
PattiPiatt	0.03	0.01	0.0	0.34	0.01	0.07	0.11	0.12	0.06	0.02	0.06	0.01	0.17
peterboghossian	0.05	0.01	0.06	0.23	0.03	0.08	0.05	0.13	0.1	0.01	0.02	0.01	0.22
rafat	0.03	0.0	0.03	0.18	0.0	0.11	0.04	0.11	0.08	0.01	0.03	0.04	0.33
realDonaldTrump	0.02	0.03	0.0	0.1	0.0	0.12	0.02	0.07	0.03	0.0	0.0	0.01	0.58
Red_or_MC1R	0.03	0.0	0.0	0.11	0.0	0.13	0.06	0.13	0.22	0.02	0.05	0.01	0.24
renormalized	0.03	0.0	0.0	0.16	0.0	0.11	0.07	0.13	0.13	0.0	0.06	0.01	0.3
RossTuckerNFL	0.03	0.0	0.22	0.2	0.05	0.09	0.09	0.13	0.04	0.01	0.02	0.02	0.11
RoxanneDawn	0.02	0.04	0.0	0.28	0.03	0.14	0.07	0.16	0.07	0.01	0.02	0.02	0.15
scimichael	0.06	0.03	0.0	0.32	0.0	0.14	0.04	0.12	0.05	0.01	0.0	0.0	0.24
SfNtweets	0.04	0.18	0.02	0.13	0.02	0.06	0.13	0.12	0.08	0.01	0.01	0.0	0.2
slatestarcodex	0.01	0.0	0.0	0.14	0.0	0.22	0.09	0.13	0.13	0.0	0.03	0.0	0.25
SLSingh	0.06	0.0	0.0	0.23	0.02	0.11	0.11	0.2	0.03	0.0	0.02	0.0	0.21
sxbegle	0.04	0.0	0.14	0.17	0.02	0.09	0.05	0.11	0.1	0.01	0.02	0.0	0.23
TeslaRoadTrip	0.05	0.01	0.0	0.23	0.0	0.12	0.06	0.1	0.04	0.01	0.02	0.03	0.34
TheMuslimReform	0.04	0.0	0.0	0.01	0.24	0.13	0.13	0.1	0.11	0.02	0.0	0.0	0.22
TheRickDore	0.01	0.02	0.22	0.1	0.0	0.12	0.03	0.11	0.08	0.02	0.05	0.02	0.2
USDISA	0.07	0.01	0.12	0.16	0.03	0.12	0.04	0.13	0.12	0.02	0.01	0.01	0.16
WestWingWeekly	0.04	0.02	0.14	0.21	0.11	0.09	0.04	0.07	0.08	0.01	0.01	0.01	0.18

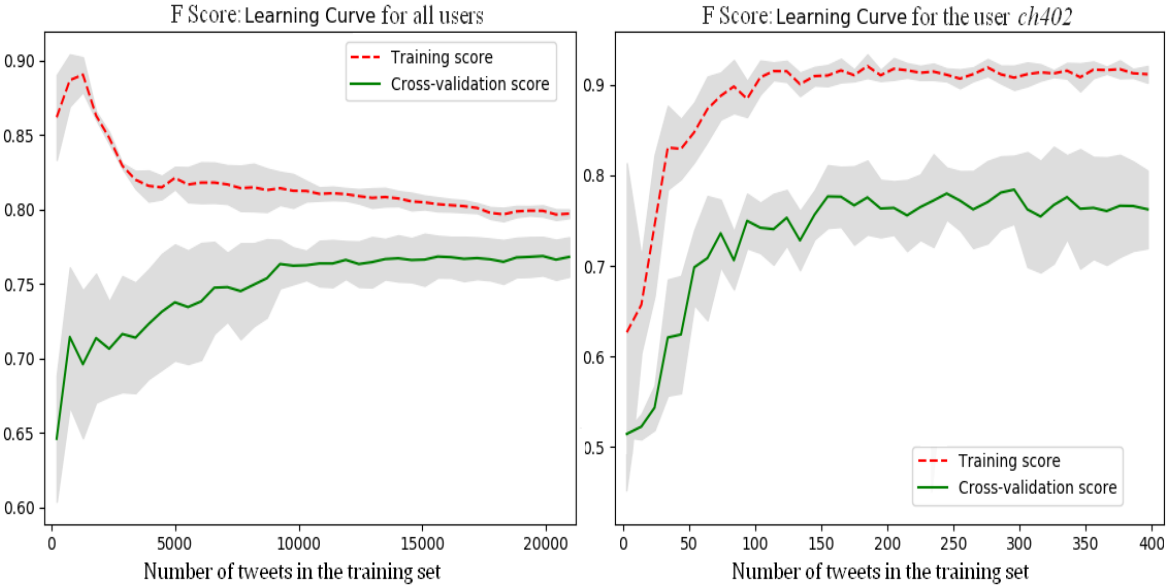

Fig. 4.5: Learning curves: Non-Personalized (left) vs. Personalized (right)

Second, the training and cross-validation curves in Fig. 4.5 indicate that both personalized and non-personalized models converge, suggesting that the models are able to learn to classify tweets according to their relevance. However, we notice that train a non-personalized model on a larger training set makes it more robust and less likely to overfit comparing to the personalized model. In other words, the training and cross-validation curves of the non-personalized model converge to the same *F score* value (76%), indicating that the model can generalize relevance predictions to unseen tweets. As to the personalized model, which is trained on a smaller training set, we observe that the model fits the training dataset too well with a high *F score* value (90%) and loses some of its ability to generalize to the cross-validation set with a lower *F score* value (78%). Therefore, to make more accurate predictions to new and unseen tweets, it would be advisable to use one of the many machine learning techniques to prevent overfitting: regularization, early stopping, data augmentation, etc. [91].

Finally, another notable difference is that non-personalized models may work better with new or inactive users, for which personalized models may have very few training instances. Indeed, in such cases, the personalized model does not have information about user preferences and interests to make specific recommendations. Hence, it is important to suggest alternatives to address this common problem in recommender systems known as the *cold-start problem*. Non-personalized models address this issue

by default since the same model can be used to any user on the social media, even new or inactive users. Lastly, it is easier for social media administrators/developers to fine-tune and manage a single non-personalized model than fine-tuning a personalized model for each user. For example, in our case, it was somewhat possible to look at each of the 46 prediction models corresponding to the 46 recipient users, but this may become more challenging as the number of users increases. In such a situation, it is necessary to provide reliable automatic techniques to validate user models.

4.6 Conclusion

In this chapter, we first presented a typical approach to rank news feed updates on *Twitter* and provided a reminder of the classical non-personalized approach that uses a prediction model for all users. Then, to predict the relevance of news feed updates and improve user experience, we used the *random forest* algorithm to train and introduce a personalized prediction model for each user. After that, we defined a fair comparison methodology by selecting the best parameters of both personalized and non-personalized models according to the collected data. Finally, we conducted a comparative study by evaluating, analyzing, and comparing personalized and non-personalized models according to six criteria: (1) the overall prediction performance of both approaches to get a global overview of the most effective model; (2) the amount of data in the training set to investigate the robustness of each model; (3) the cold-start problem, which is a common problem in recommender systems; (4) the incorporation of user preferences over time; (5) the model fine-tuning to investigate the manageability of each model; and (6) the personalization of feature importance for users. Following extensive experiments on a dataset crawled from *Twitter*, the experimental results show that a single non-personalized model for all users is easy to manage and fine-tune, is less likely to overfit as it benefits from more data, and it addresses the problem of cold-start and inactive users. On the other hand, the personalized models we introduce allow personalized feature importance, take into consideration the preferences of each user, and allow to track changes in user preferences over time. Furthermore, the personalized models we propose give a higher prediction accuracy than non-personalized models. These findings highlight the need for personalization to effectively rank the news feed, promote relevant updates, and assist users by suggesting tailored content of interest.

In the next chapter, knowing that the effectiveness of the prediction and ranking

depends partly on the chosen model, we first describe the context of the comparison on *Twitter* according to a personalized approach that predicts the relevance of news feed updates. Then, we select and describe seven supervised learning algorithms that have been used in related work. Finally, to determine the most suitable models, we conduct a comparative study by evaluating, analyzing, and comparing the selected algorithms.

Chapter 5

Supervised learning algorithms for personalized RNFU: A comparative study

5.1 Introduction

In several research approaches, ranking news feed updates in descending relevance order has been proposed to help users quickly catch up with the content they may find interesting [55]. For this matter, different supervised learning models such as *Gradient Boosting*, *Support Vector Machine*, and *logistic regression* have been commonly used in related work and seem suitable to rank news feed updates [93]. Indeed, using labeled training data, these models analyze past user behaviors to predict whether they will find an update relevant in the future [91]. The purpose is to map new input features of an update unread by a user to a relevance score using a function learned from previously read updates in the training set. However, we found that each related work intuitively chooses one supervised learning algorithm, states that the other algorithms can be used, and points out that it is out of the scope to compare different algorithms in their work [94]. According to the *no free lunch* theorem [36], no single machine learning algorithm is better than all the others on all problems as each one of them has its advantages and disadvantages such as flexibility, complexity, interpretability, overfitting tendency, times for learning and predicting, tolerance to a large number of features, minimal required data, number of hyperparameters, etc. Therefore, it is common to try multiple models and select the one that works best for a particular

problem. The key to a fair comparison of machine learning algorithms is ensuring that each algorithm is evaluated in the same way on the same data [36].

In this chapter, knowing that the effectiveness of the prediction and ranking depends partly on the chosen model, we first describe the context of the comparison on *Twitter* according to a personalized approach that predicts the relevance of news feed updates. Then, we select and describe seven supervised learning algorithms that have been used in related work: *Naive Bayes*, *logistic regression*, *decision trees*, *Gradient Boosting*, *random forest*, *artificial neural networks*, and *Support Vector Machine*. After that, we define a rigorous and fair comparison methodology by selecting the best parameters of each algorithm according to the data. Finally, to determine the most suitable models, we conduct a comparative study by evaluating, analyzing, and comparing the selected algorithms. The comparison and evaluation results are presented and discussed according to three criteria: (1) overall prediction performance of all algorithms to get a global overview of the most effective models; (2) prediction performance on various training set sizes to investigate scalability and the impact of data size on model performance; and (3) the computing speed performance to have an insight into the fastest models for eventual deployment in production.

The chapter is structured as follows: section 5.2 describes the context of the comparison on *Twitter* according to a personalized approach that predicts the relevance of news feed updates, section 5.3 describes the seven supervised learning algorithms that we selected for the comparison, section 5.4 discusses the experiments we performed to evaluate and compare the supervised models, and section 5.5 concludes the chapter.

5.2 Context of the comparison

Fig. 5.1 describes the personalized technique we introduced in the chapter 4 to predict the relevance score $\boldsymbol{R(t,u)}$ of a tweet $\boldsymbol{t \in F(u)}$. This technique is based on a supervised prediction model that analyzes labelled training data of tweets that $\boldsymbol{u}$ read in the past to predict if $\boldsymbol{u}$ will find $\boldsymbol{t}$ relevant in the future. Let $\boldsymbol{D(u)}$ denotes a subset of tweets previously read by $\boldsymbol{u}$. The training data is a set of input-out pairs such that an input represents a vector of features that may influence the relevance of a tweet $\boldsymbol{t' \in D(u)}$ to $\boldsymbol{u}$, and the output represents the relevance score $\boldsymbol{R(t',u)}$. The technique involves three steps: (1) assign implicit relevance scores to tweets; (2) extract the features that

may influence relevance; and (3) train the relevance prediction model. In this section, we describe each step according to our personalized approach.

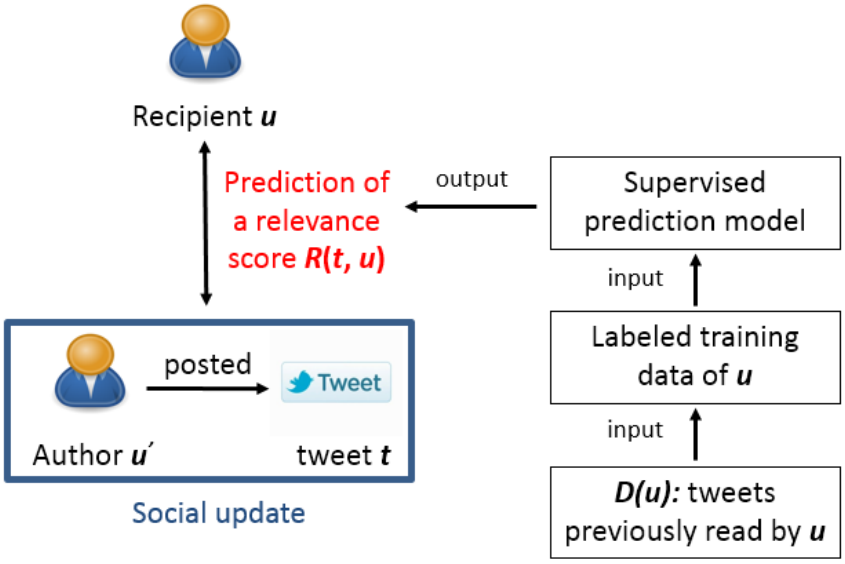

Fig. 5.1: Personalized prediction of a relevance score

5.2.1 Relevance scores

The implicit method we use assumes that a previously read tweet $t' \in D(u)$ is relevant to a user $u \in S$ if u interacted with the tweet t'. Predicting implicit relevance scores results in a binary classification problem:

$$R(t',u) = \begin{cases} 1 & \text{if } u \text{ interacted with } t' \text{ (retweet or reply or like)} \\ 0 & \text{otherwise} \end{cases} \tag{5.1}$$

5.2.2 Features that may influence relevance

We use the 13 most relevant features according to related work, which were previously presented in section 4.2.2. The features may influence the relevance $R(t,u)$ of a tweet t, posted by an author u', to the recipient u. We recall the features in Table 5.1.

5.2.3 Relevance prediction model

Let S denotes the set of recipient users. First, we generate training data instances for each user $u \in S$ in the form of input-output pairs considering each previously read

Table 5.1: Features that may influence relevance

Features that may influence relevance		**Type**	**N°**
Relevance of the content of ***t***, its hashtags, and mentions to ***u***	Relevance of the keywords of ***t*** to ***u***	Int	*f1*
	Relevance of the hashtags of ***t*** to ***u***	Int	*f2*
	Presence of ***u*** in the mentions of ***t***	Bool	*f3*
Social tie strength between ***u*** and ***u'***	Interaction rate of ***u*** with tweets of ***u'***	Float	*f4*
	Number of times ***u*** mentioned ***u'***	Int	*f5*
Authority of ***u'***	Followers count / Followings count	Int	*f6*
	Seniority in years	Int	*f7*
	Listed (group) count	Int	*f8*
Quality of ***t***	Length (# characters)	Int	*f9*
	Presence of hashtags	Bool	*f10*
	Presence of a URL	Bool	*f11*
	Presence of a photo or a video	Bool	*f12*
	Popularity (# retweets, replies, likes)	Int	*f13*

tweet ***t'*** $\in$ ***D(u)***. An input represents a vector of features that may influence the relevance of ***t'*** to ***u***, and the output represents the implicit relevance score ***R(t',u)***. Second, we divide the training data of each user ***u*** $\in$ ***S*** into two sets: a training set of the prediction model for 70% of the first instances (the least recent ones) and a test set for 30% of the remaining instances (the most recent ones). Finally, we use the training set of each user ***u*** $\in$ ***S*** to train a supervised prediction model. The purpose is to map new input features of a tweet unread by ***u*** to a relevance score using a binary classifier learned from previously read tweets in the training set. Different supervised algorithms have been used in related work to train the relevance prediction model. In the next section, we describe the selected algorithms that we consider in the comparison.

5.3 Selected supervised learning algorithms

In the comparison, we first selected the following seven supervised algorithms that have been used in several related works: *Naive Bayes* [7], *logistic regression* [8, 9, 55], *decision trees* [95], *Gradient Boosting* [5], *random forest* [83], *artificial neural networks* [66, 67, 54], and *Support Vector Machine* [56]. In this section, we describe each of the algorithms briefly and refers to the corresponding related works that have used them. Note that *Naive Bayes* and *logistic regression* can be used for classification only, unlike the five other algorithms mentioned above, which can be used for both classification and regression problems. However, as discussed in section 5.2.1, we recall that predicting relevance scores of tweets is a binary classification problem. The rest of the section therefore focuses on supervised classification.

5.3.1 Naive Bayes

In an attempt to improve the performance of the state-of-the-art collaborative personalized tweet recommendation model [64], Song et al. [7] introduced personal hashtags to optimize BPR (Bayesian Personalized Ranking) [96] and creatively apply the model into an online scenario. The Bayes theorem with independent assumptions between features is the core concept of the *Naive Bayes* classifier [34]. The simplest approach of the Bayesian network is *Naive Bayes*, in which all features of a dataset are independent of the class variable value. Therefore, the *Naive Bayes* classifier can be considered a Bayesian network where the class node has no parents, and each feature node has the class as its sole parent. *Naive Bayes* classifiers build the model easily with no complicated iterative parameter estimation [42]. Due to this characteristic, *Naive Bayes* classifiers are useful for large datasets. Despite its simplicity, *Naive Bayes* classifiers provide better results for complex real-world problems. Indeed, several studies on the analysis of the Bayesian classification problem have shown that there are some theoretical reasons for the unreasonable effectiveness of *Naive Bayes*, even on datasets with substantial feature dependencies [34]. One of the strengths of *Naive Bayes* is that it requires a small amount of training data to estimate the target function [42].

5.3.2 Logistic Regression

To predict the probability that a recipient user may find a tweet interesting enough to retweet it, Vougioukas et al. [55] proposed a binary classifier based on *logistic regression*.

Similarly, to personalize news feeds on *LinkedIn*, Agarwal et al. [8, 9] proposed large-scale binary classifiers based on *logistic regression* to predict the click-through-rate of each activity and generate three kinds of affinity scores: viewer-activity, viewer-actor, and viewer-activity-actor. The term regression is defined as analyzing or measuring the relationship between a dependent variable and one or more independent variables [40]. Regression is defined by two types: *linear regression* and *logistic regression*, where *logistic regression* is a generalization of *linear regression* such that the continuous output variable is transformed into a discrete variable [97]. Indeed, *logistic regression* is used to estimate binary or multi-class dependent variables where a boundary between the classes exists. It is therefore used to classify the low-dimensional data having non-linear boundaries. *Logistic regression* states that the class probabilities depend on the distance from the boundary [34]. It also provides the difference in the percentage of the dependent variable and provides the rank of features according to their importance. *Logistic regression* is fast to train, seldom overfits, and is commonly used for applied statistics, discrete data analysis, and event probability prediction [30].

5.3.3 Decision Trees

In an effort to predict the relevance of news feeds updates and leverage the author's expertise in addition to other features used in related work, we proposed in [95] a binary classifier model based on *decision trees*. *Decision trees* are flowchart-like structures that classify instances by sorting them based on feature values [41]. In an instance to be classified, each node in a *decision tree* represents a test on a feature, each branch represents the output of the test, i.e. a value that the node can have, and each leaf node represents the class label. Instances are classified starting at the root node and sorted based on their feature values until the leaf nodes [98]. The paths from the root to the leaf represent relevance classification rules. In the decision process, the instances are split into two or more sub-sample sets, which is decided by the most significant splitter or differentiator in the input features [41]. *Decision trees* are fast to train and to predict outcomes, they require little data preprocessing, and they are easy to interpret with their visual representation, unlike most models that are black boxes [98].

5.3.4 Gradient Boosting

To predict continuous relevance scores for tweets, Shen et al. [5] first modeled the relevance of tweets by minimizing the pairwise loss of relevant and irrelevant tweets.

Then, they used a supervised regression model based on a *Gradient Boosted ranking* algorithm (GBrank) [68] to learn a tweet ranking function. *Gradient Boosting* is an *ensemble learning* method[1] for regression and classification problems that produce a prediction model in the form of an ensemble of weak prediction models, typically *decision trees* [34]. *Gradient Boosting* combines the weak learners into a single strong learner in an iterative fashion. The algorithm begins by initializing the ensemble with a single weak model whose predictions can be pretty naive, then start the cycle. As each weak learner is added, a new model is fitted to provide a more accurate estimate of the output variable. The new weak learners are maximally correlated with the negative gradient of the loss function associated with the whole ensemble [34]. *Gradient Boosting* can approximate most nonlinear functions; it also addresses concerns about multicollinearity problems, where there are high correlations between features [34]. *Gradient Boosting* has shown success in practical applications and various machine learning and data mining challenges.

5.3.5 Random Forest

To compute feature importance and predict the relevance of news feed updates on *Twitter*, we proposed in [83] a supervised model based on *random forest* to predict binary scores for tweets. *Random forest* is an ensemble learning method for classification and regression problems that operate by constructing a multitude of *decision trees* called *estimators*, which each produce their own predictions [43]. Each tree is distinguished by the sub-sample vector on which it is trained, and which is randomly selected with the same distribution from the training set. This machine learning process is called *bagging* [34]. *Random forest* uses averaging, i.e. takes the average of the predictions of the individual estimators to improve the predictive performance and correct the decision trees' habit of overfitting [43]. The *random forest* algorithm has been extremely successful as a classification and regression method in a wide range of prediction problems such as data science, bioinformatics, 3D object recognition, etc. [99]. *Random forest* is extremely robust, especially in the classification, it seldom overfits, implicitly performs feature selection, and allows to compute feature importance [89].

[1] A method that uses multiple machine learning algorithms to obtain better predictive performance than could be obtained from any of the constituent learning algorithms alone.

5.3.6 Artificial Neural Networks

To represent tweets, user preferences, compute similarities between user preferences and tweets, and rank tweets according to their relevance, Zhang et al. [66], Piao and Breslin [67], and De Maio et al. [54] proposed different approaches based on *artificial neural networks*. The general architecture of a neural network consists of three layers: an input layer that defines the input value, one or more hidden layers that define the mathematical function, and an output layer that defines the final outcome [33]. Each layer consists of a large number of neurons that are interconnected through weights such that each neuron has a mathematical function, called *activation function*, that takes input from the previous layer and produces output for the next layer [30]. The behavior of the ANN is defined by the values of the weights. Indeed, the weights of the neural network to be trained are initially set to random values, then instances of the training set are repeatedly exposed to the network as follows: First, the values of an input instance are placed on the input units. Then, the output of the network is compared to the desired output for this instance. Finally, all the weights in the network are adjusted slightly in the direction that brings the output values of the network closer to the values for the desired output [33]. *Artificial neural networks* are applied to many real-life problems; one of their strengths is their robustness to outliers and their ability to approximate any nonlinear function from data [34].

5.3.7 Support Vector Machine

To predict the relevance of news feed posts on *Facebook*, Paek et al. [56] learned *Support Vector Machine* classifiers of news feed importance to identify predictive features related to social media properties, the message text, and the user background information. In *support vector machine*, each data point is interpreted as a p-dimensional vector, such that the machine attempts to create a linear classifier by fitting the data point inside a hyperplane with a p-1 dimension [44]. A hyperplane in an n-dimensional Euclidean space is a flat n-1 dimensional subset of that space that divides it into two separate parts [30]. Indeed, every input is turned into a point in n-dimensional space where n is the number of features, and the value of each feature is defined as the value of a unique coordinate on the hyperplane [34]. The classification is determined by finding the hyperplane that most clearly separates the two classes [44]. Maximizing the margin separating the support vectors and creating the largest distance between the hyperplane and the instances on either side has been proven to reduce an upper bound

on the generalization error [44]. Unlike *logistic regression*, the *support vector machine* does not provide class probabilities but outputs a class identity. The key strength of *support vector machines* is their ability to approximate complex non-linear functions and to work well when there is a clear margin of separation between classes [34].

To the best of our knowledge, no comparative study was carried out to determine the most suitable models as each related work intuitively chooses one supervised learning algorithm without addressing the possible use of other algorithms that can be more efficient. The *no free lunch* theorem [36] for supervised machine learning is a theorem that essentially implies that no single machine learning algorithm is universally the best-performing algorithm for all problems. Indeed, each algorithm has its advantages and disadvantages such as flexibility, complexity, interpretability, overfitting tendency, times for learning and predicting, tolerance to a large number of features, minimal required data, number of hyperparameters[2], etc. Hence, it is common to try different models and select the one that works best for a particular problem using a validation technique. In the next section, knowing that the effectiveness of the prediction depends partly on the chosen model, we describe the experiments we performed to evaluate, compare, and determine the most suitable algorithms to rank news feed updates.

5.4 Experiments and comparison results

To evaluate and compare the supervised models, we describe in this section: (1) the measures we used to evaluate model performance; (2) the methodology we used in the comparison; and (3) the obtained results.

5.4.1 Measures

In the experiments, we used the dataset described in section 4.5.1 and the concepts TP, TN, FP, and FN defined in section 4.5.2. To compare model performance in predicting the relevance of news feed updates, we first train a binary classifier model for each user $\boldsymbol{u} \in \boldsymbol{S}$ using the corresponding training set (70% of the least recent instances). Then, we evaluate the model using the corresponding test set (30% of the most recent instances). To do so, we use two popular measures for binary classification, the *Accuracy* given by Equation 5.2 and the weighted *F1 score* given by Equation 5.3 [91].

[2]The "knobs" to be tweak during successive runs of training a model.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{5.2}$$

$$F = \frac{(F_r \times (TP + FN)) + (F_i \times (TN + FP))}{TP + TN + FP + FN} \tag{5.3}$$

Where:

- F_r is the standard *F1 score* for the class of relevant tweets
- F_i is the standard *F1 score* for the class of irrelevant tweets

5.4.2 Methodology

The experimentation was conducted using *Python* and the *scikit-learn* machine learning library[3], which includes various classification, regression, and clustering algorithms.

First, we annotated the supervised algorithms as follows: *Naive Bayes* (NB), *logistic regression* (LR), *decision trees* (DT), *Gradient Boosting* (GB), *random forest* (RF), *artificial neural networks* (ANN), and *Support Vector Machine* (SVM). Then, since ANN and SVM require data scaling to prevent features in greater numeric ranges, we normalized all feature values in the range [0,1] using the min-max scaling [100]. Moreover, for a fair comparison, we selected the best parameters of each algorithm with a *random search* [92] and a 5-fold time-series cross-validation performed on the train set [91]. A time-series cross-validation was used instead of the standard cross-validation to preserve the chronological order of tweets (see section 5.2.3).

To find the best parameters of each algorithm, a *random search* was hence run over different parameter values. *Random search* is a widely used strategy for algorithm hyper-parameter optimization that randomly samples values from the statistical distribution of each parameter to select the values that give the best performance according to the evaluation measure [92]. Table 5.2 gives details about the model hyperparameter optimization, including the distribution of the most important parameters for each algorithm and the number of iterations we performed in the corresponding *random search*. The description of the hyperparameters of each algorithm is provided in Table 5.3, while more details are available in the *scikit-learn* documentation[3]. The

[3]https://scikit-learn.org/stable/

Table 5.2: Model hyperparameter optimization

Supervised model	Parameter distribution	Nº of iterations in *random search*	Related work
GB	loss: [deviance, exponential] learning_rate: [0.001, 0.01, 0.1] n_estimators: [15, 30, 50, 70, 100] subsample: [0.5, 0.7, 1.0] min_samples_split: [4, 7, 10, 13, 16, 19] min_samples_leaf: [3, 7, 10, 13, 16, 19] max_depth: [3, 6, 9] max_features: [sqrt, log2]	150	[5]
RF	n_estimators: [15, 30, 50, 70, 100] criterion: [entropy, gini] max_depth: [3, 6, 9] min_samples_split: [4, 7, 10, 13, 16, 19] min_samples_leaf: [3, 7, 10, 13, 16, 19] max_features: [sqrt, log2] bootstrap: [True, False]	150	[83]
SVM	C: [1, 10, 100, 1000] kernel: [linear, rbf, poly] gamma: [0.0001, 0.001, 0.1, 1]	48	[56]
DT	criterion: [entropy, gini] max_depth: [3, 6, 9] min_samples_split: [4, 7, 10, 13, 16, 19] min_samples_leaf: [3, 7, 10, 13, 16, 19]	150	[95]
ANN	optimizer: [SGD] input neurons: [13] hidden neurons: [13, 30] output neurons: [1] epochs: [15, 30] learn_rate: [0.1, 0.2, 0.3] momentum: [0.2, 0.4, 0.6] dropout_rate: [0.1, 0.2, 0.3]	54	[66] [54] [67]
LG	penalty: [l1, l2] dual: [False] C: [0.001, 0.01, 0.1, 1, 10, 100, 1000] solver: [newton-cg, lbfgs, liblinear, sag, saga]	70	[8] [9] [55]
NB	-	-	[7]

number of iterations we use is approximately chosen according to the number of parameters of each algorithm and their distribution, i.e. the more parameters an algorithm has and the denser their distribution, the greater the number of iterations to perform. Note that ANN is an exception as it has many parameters to fine-tune, but a single iteration is very computationally expensive.

Table 5.3: Hyperparameter description [101]

Hyperparameter	**Description**
loss	The loss function to be optimized
learning_rate	The step size at each iteration while moving toward a minimum of a loss function
n_estimators	The number of trees in the ensemble model
subsample	The fraction of samples to be used for fitting the individual trees
min_samples_split	The minimum number of samples required to split an internal node in a tree
min_samples_leaf	The minimum number of samples required to be at a leaf node in a tree
max_depth	The maximum depth of a tree
max_features	The number of features to consider when looking for the best split of an internal node in a tree
criterion	The function to measure the quality of a node split in a tree
bootstrap	Whether bootstrap samples are used when building trees. If False, the whole dataset is used to build each tree
C	The regularization parameter to reduce overfitting. The strength of the regularization is inversely proportional to C
kernel	The kernel type to be used in the algorithm
gamma	The kernel coefficient, which defines how far the influence of a single training example reaches
optimizer	The training optimization algorithm to be used. The most common algorithm is Stochastic Gradient Descent (SGD)
input neurons	The number of neurons in the input layer
hidden neurons	The number of neurons in the hidden layer
output neurons	The number of neurons in the output layer
epochs	The number of iterations. One iteration is generally defined as one pass over the entire dataset
momentum	A value that controls how much to let the previous weight updates influence the current weights
dropout_rate	The proportion of neurons to be ignored during training to reduce overfitting
penalty	The norm used in the penalization to penalize the logistic model for having too many variables
dual	A parameter to use dual or primal formulation, Dual=False is preferred when n_samples >n_features
solver	The training optimization algorithm to be used

Finally, to study the algorithmic stability with several runs and small changes to training data, we retrained and iterated each model on 20 different *random state*[4] values, then evaluated it on the test set. Therefore, we end up with 20 *Accuracy* and *F scores* for each algorithm and then plot the corresponding boxplot and average scores.

[4]A variable used in randomized algorithms to determine the random seed of the pseudo-random number generator.

5.4.3 Results

The comparison and evaluation results are presented and discussed according to three criteria: (1) overall prediction performance of all algorithms to get a global overview of the most effective models; (2) prediction performance on various training set sizes to investigate scalability and the impact of data size on model performance; and (3) the computing speed performance to have an insight into the fastest models for eventual deployment in production.

Overall performance Fig. 5.2 shows the comparison between the average *Accuracy* and *F scores* for each algorithm, and Fig. 5.3 presents the boxplot of the comparison between the algorithms trained and evaluated over 20 *random state* values. Fig. 5.2 indicates that the *Accuracy* scores are slightly higher than the *F scores*. Indeed, in our dataset, the classes are unbalanced since the relevant tweet class represents only about 35% of tweets for each user. The *Accuracy* measure gives slightly over-optimistic results comparing to the *F score* because the latter is weighted by the number of true instances for each class, unlike the *Accuracy* score. Therefore, we focus on the *F score* in the rest of the section. First, both figures show that the scores differ greatly depending on the algorithm, from average *F scores* of 80.7% and 80.68% with GB and RF respectively, to 73.29% with NB. This corresponds to a performance gain of almost +8% and confirms that comparing and selecting the most suitable supervised model is critical to increase the efficiency of the prediction and ranking process. Moreover, we note from Fig. 5.3 that SVM, LR, and NB remain stable even if retrained and iterated over different *random state* values. Indeed, the learning process of these models does not imply random sampling from data, unlike GB, RF, DT, and ANN [91].

Nonetheless, although GB and RF are less stable, we notice from Fig. 5.2 and Fig. 5.3 that they outperform the other algorithms, with average *F scores* of 80.7% and 80.68% respectively. This highlights that ensemble learning models such as GB and RF, which combine multiple learning algorithms to improve the overall performance, are the most appropriate to predict the relevance of news feed updates. Moreover, this confirms our intuition in the chapter 4 in which we used *Random Forest* because of its many advantages. Note that both GB and RF combine multiple *decision trees*. The results also reveal that SVM and DT perform well with average *F scores* of 77.49% and 77.63% respectively, but not as well as GB and RF. Indeed, both GB and RF combine multiple *decision trees* to improve the predictive performance and correct the overfitting

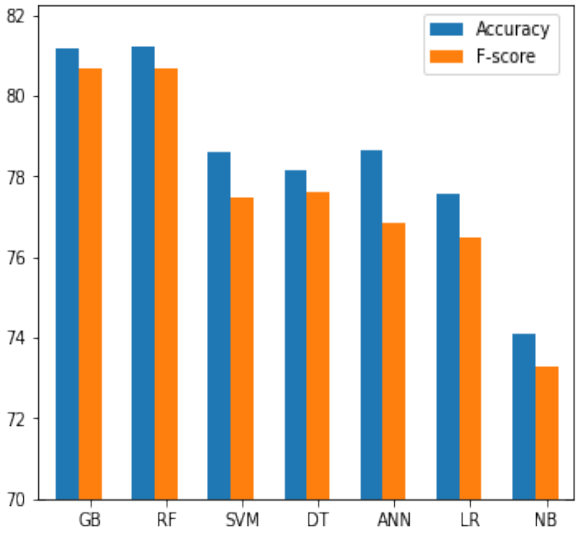

Fig. 5.2: Average *Accuracy* and *F scores*

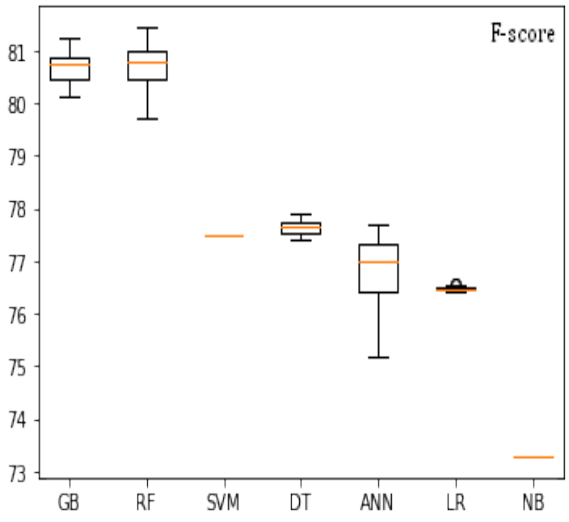

Fig. 5.3: *F scores* on 20 *random state* values

tendency of the individual DT model, which tends to result in a long tree and loses its ability to generalize to unseen data [89]. Regarding the SVM model, as in the case study of ranking news feed updates, it has been proved that ensemble learning methods such as GB and RF tend to perform better than SVM in clearly defined problems with tabular data[5], small to intermediate datasets, and a manageable number of features [91].

The results also indicate that ANN and LR perform moderately with average *F scores* of 76.86% and 76.48% respectively. Indeed, ANN is computationally expensive, and its architecture has many parameters that are difficult to fine-tune (number of layers, activation function and number of neurons in each layer, etc.) [33]; the *random search* hyper-parameter optimization we used could not cover the statistical distribution of all parameters in the large search-space (see Table 5.2). Moreover, ANN requires a considerable amount of data to achieve a good performance; in this case of study, there is only an average of 569 tweets as training data instances for each user. Finally, neural networks tend to perform very well with unstructured data such as text and images, but not with tabular data such as our dataset [102]. Concerning LR, it is a parametric machine learning algorithm since it assumes a linear functional form of the prediction model [40]. Parametric algorithms such as LG are simpler as they make

[5]In machine learning, data can be categorized into unstructured data, which can be maintained in formats that are not uniform, such as image and text, and structured data with the common tabular rows and columns format.

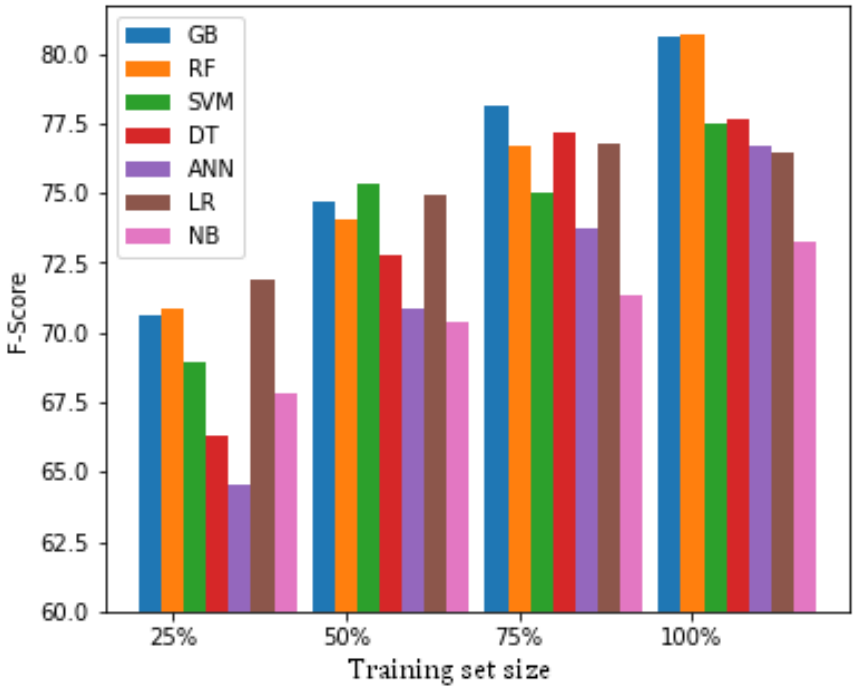

Fig. 5.4: Average *F scores* on various training set sizes

assumptions about the target function to make it easier to learn; they are faster to train, require less data but are not as powerful as GB, RF, SVM, and DT, which are non-parametric algorithms [103]. Non-parametric methods are more flexible as they can learn any functional form from the training data, but have a higher model complexity and require more data and training time [30]. Finally, we observe that NB performs poorly with an average *F score* of 73.29%. This is certainly due to the strong assumption this parametric algorithm makes about the independence of the features, which do not hold in the case study of ranking news feed updates. Indeed, the value of some features is dependent on the value of other features, e.g., the more followers a user has (feature f_6), the more the user is listed in groups (feature f_8).

Performance on various training set sizes To investigate scalability and the impact of data size on performance, we computed in Fig. 5.4 the average *F scores* on various training set sizes: 25%, 50%, 75%, and 100% of the data. For each training set size, we retrained and iterated each of the seven algorithms over 20 *random state* values. First, the results highlight the huge impact of data size on the performance of all algorithms as we notice a significant gain when using 100% of data compared to 25% of data, e.g. from an average score of 70.6% to 80.57% with GB, from 70.89% to

80.69% with RF, from 68.93% to 77.49% with SVM, etc. The results also indicate that the ranking of the most effective algorithms changes as we consider larger training set sizes. For example, at first, when considering only 25% and then 50% of data, LR is the first and then the second most efficient algorithm with average scores of 71.88% and 74.96% respectively. Similarly, when considering 25% of data, we notice that NB outperformed ANN and DT with an average score of 67.8% for NB against 64.57% and 66.32% for ANN and DT. Indeed, when the training set is small, parametric classifiers such as LR and NB, which have a high bias/low variance[6], have an advantage over non-parametric classifiers such as GB, RF, SVM, ANN, and DT, which have a low bias/high variance. Indeed, due to their flexibility in learning any functional form from the training data, non-parametric classifiers tend to overfit small training sets and lose their ability to converge and generalize to unseen data [30]. However, as the training set grows with 75% and then 100% of data, we notice that low bias/high variance classifiers start to win out so that LG moved from the second to the fourth and then to the fifth most efficient algorithm win an average score of 76.74% and then 76.48% respectively. Indeed, the increasing amount of data and its complexity fails to meet the simplifying assumptions and low-model complexity of high bias classifiers such as LG and NB, unlike low bias classifiers such as GB, RF, SVM, ANN, and DT, which start to converge as training size increases [103].

Speed performance To compare computing speed, we calculated in Fig. 5.5 and Fig. 5.6 the average training and predicting times for each of the seven algorithms over 20 executions. In both training and predicting, a single run corresponds to the algorithm execution time for all 46 users in the dataset, which corresponds to 18307 tweets for training and 7873 tweets for predicting. First, both figures show that the fastest algorithm is DT with average training and predicting times of 0.04 and 0.0 seconds respectively. Indeed, one of the specifications of *decision trees* is to be very quick to train and predict outcomes thanks to their flow-chart structure [98]. The second fastest algorithms are NB and LR with average training and predicting times of 0.03 and 0.01 seconds for NB and 0.83 and 0.01 seconds for LR. Certainly, as they make strong assumptions about the form of the learning function to make it easier to learn, the simplicity and low model complexity of parametric algorithms such as NB and LG make them faster to train and predict outcomes than non-parametric algorithms [103].

[6]The bias is the simplifying assumptions made by a model to make the target function easier to learn, and the variance is the amount of changes in the target function if different training data was used. The goal of machine learning algorithms is to achieve low bias and low variance.

The results also indicate that SVM, GB, and RF are relatively fast in both training and predicting with average training and predicting times of 1.28 and 0.08 seconds for SVM, 1.93 and 0.02 seconds for GB, and 2.84 and 0.13 seconds for RF. Indeed, these algorithms give overall good speed performance as long as the dataset is not too large [30], which is the case in our study. Finally, the results show that ANN is extremely slow in training and predicting than the other models with average training and predicting times of 51.57 and 2.72 seconds. Indeed, due to its complexity and model architecture, ANN is much more computationally expensive and time-consuming to train and predict outcomes than traditional algorithms, even with small to intermediate datasets [33].

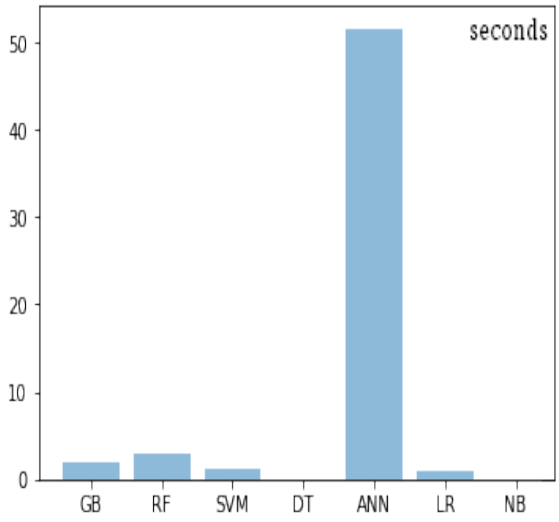

Fig. 5.5: Average training time on 18307 tweets

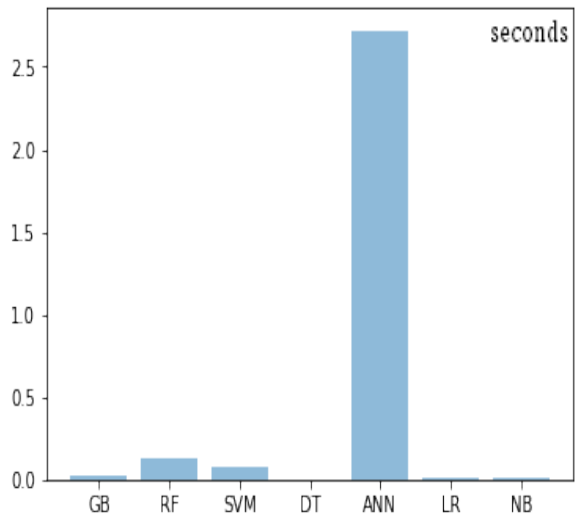

Fig. 5.6: Average predicting time on 7873 tweets

5.5 Conclusion

In this chapter, we first described the context of the comparison on *Twitter* according to a personalized approach that predicts the relevance of news feed updates. Then, we selected and described seven supervised algorithms that have been used in related work to predict the relevance. After that, we defined a rigorous and fair comparison methodology by selecting the best parameters of each algorithm according to the data. Finally, to determine the most suitable models, we conducted a comparative study by evaluating, analyzing, and comparing the selected algorithms according to three

criteria: (1) overall prediction performance of all algorithms to get a global overview of the most effective models; (2) prediction performance on various training set sizes to investigate scalability and the impact of data size on model performance; and (3) the computing speed performance to have an insight into the fastest models for eventual deployment in production. Following extensive experiments on *Twitter*, the results highlight that selecting the most suitable supervised model is critical to increase the efficiency of the ranking process. Furthermore, the results show that ensemble learning models such as *Gradient Boosting* and *Random Forest* are the most appropriate to predict the relevance of updates. Indeed, the comparison results showed that these models are fast, scalable, and outperform the other algorithms in prediction. Moreover, they become very accurate as the data size increases.

In the next chapter, we first analyze and discuss research work in the field of expert finding on social media. Then, to investigate the contribution of expertise to rank news feed updates, we propose a personalized approach that predicts the relevance using supervised prediction models based on *random forest*. In addition to the features used in related work, the proposed approach leverages the author's expertise that we infer from the biography and the textual content the author has posted.

Chapter 6

Personalized expertise-aware approach for RNFU

6.1 Introduction

As mentioned in the previous chapters, in most approaches, ranking news feed updates in descending relevance order has been achieved based on the prediction of a relevance score between a recipient user and a new update in the news feed [9]. These approaches generally use four types of features that may influence relevance [104]: (1) the relevance of the update content to the recipient's interests; (2) the social tie strength between the recipient and the author; (3) the author's authority; and (4) the update quality. We believe that using these features is necessary when predicting the relevance but not sufficient. For example, updates on a specific topic authored by a novice user may not attract the attention as much as updates posted by a recognized expert in his field. Indeed, updates posted by experts, also known as topical influencers [77] or topical authorities [105], are often considered credible, important, and interesting [78]. Therefore, leverage the author's expertise could be all the more necessary to enable recipient users to catch up with the valuable and trustworthy updates on specific topics.

Expertise information is usually not explicitly provided by social media users [80]. Therefore, existing research methods primarily rely on implicit expert finding, which aims at identifying users with the relevant knowledge or experience on a given topic [86]. The expert finding is a critical problem that has been broadly studied in many applications such as viral marketing, recruiting talent, link and user recommendation, and question answering [76]. The task of finding topical experts in networked data

has been traditionally addressed by combining text mining with link analysis through the network structure [53]. As regards social media, the main techniques used to infer a user's expertise leverage other user interactions with the textual content the user has posted [106] as well as the user's social behaviors, including user-generated content, biographical information, social relationships, list memberships[1], etc. [80]. Based on existing work, to the best of our knowledge, the author's expertise has not been used before when recommending and predicting the relevance of updates to recipient users. In this chapter, we propose a personalized approach to investigate the contribution of expertise to rank news feed updates. In addition to other features used in related work, the proposed approach leverages the author's expertise that we infer from the biography and the textual content the author has posted. The experimental results on *Twitter* show that the features and *random forest* models we used overall succeed in predicting the relevance of tweets with remarkable results for several recipient users. Moreover, the results show that infer and introduce the author's expertise has improved the classical approach with a considerable gain in prediction for many users. The feature importance also shows that expertise is ranked among the top features when judging the relevance of tweets by recipients users.

The chapter is structured as follows: section 6.2 provides an analysis on expert finding on social media, section 6.3 describes the proposed approach which uses the author's expertise in addition to other features, section 6.4 presents the experiments we performed to evaluate our approach, and section 6.5 concludes the chapter.

6.2 Expert finding

To identify and find experts on social media, different research approaches have been proposed in the expert finding field. According to the research work we presented in section 3.5.1, we notice that the biography and tweets posted by a user have been broadly used to accurately infer the user's expertise. Indeed, these two user-related data are complementary. On the one hand, self-reported biographies may indicate explicit information about expertise such as education, skills, career information, etc. On the other hand, the tweets authored by a user may indicate implicit information about the expertise. The assumption is that a user is likely to have expertise in topics on which he frequently expresses his opinion. We believe that these findings can be

[1]Lists that allow users to organize people they follow into labeled topical groups.

leveraged to predict the relevance of tweets to recipient users. Indeed, as discussed in section 3.4.1, the primary technique used to predict a relevance score to a tweet ***t*** is based on a relevance prediction model that uses as input a vector of features that may influence the relevance of ***t*** to the recipient user ***u***, to output a corresponding relevance score ***R(t,u)*** that measures the relevance of ***t*** to ***u***. Four types of features that may influence relevance were used in related work:

- Features that measure the relevance of the textual content of ***t*** to the interests of ***u***. Certainly, the features that match between the tweet content and the recipient's interests (inner product, cosine similarity, common words, etc.) may serve as direct predictors of relevance.
- Features that measure social tie strength between ***u*** and ***u'***: interaction rate, number of common friends, etc. The assumption is that ***t*** could be relevant to ***u*** if ***u*** has a strong social relationship with ***u'***.
- Features that measure the authority of ***u'***: followers and followings count, seniority, etc. The assumption is that ***t*** could be relevant to ***u*** if ***u'*** has authority on the social media. Nagmoti et al. [69] state that if a user is important, i.e. has authority, then his tweets are also important.
- Features that measure the quality of ***t***: length, popularity, the presence of multimedia content, etc. The assumption is that ***t*** could be relevant to ***u*** if it is of high quality (long, popular, formal, informative, etc.).

Based on existing work, the features that measure the expertise of ***u'*** in the topics of ***t*** have not been used when recommending and predicting the relevance of updates. The assumption is that the tweet ***t*** could be relevant to the recipient ***u*** if the author ***u'*** is an expert in the topics of ***t***. According to Wagner et al. [78], tweets posted by experts, i.e. users with the relevant skill or knowledge on a given topic, are often considered credible, important, and interesting. For example, *Elon Musk* is very famous in the tech culture for his warnings about the risks of Artificial Intelligence, and his tweets on this subject often attract the attention of users[2] (see Fig. 6.1). Indeed, unlike novice users, experts know what they are talking about when it comes to topics they master [79]. Therefore, identifying experts could be crucial to allow users to catch up with the valuable tweets on specific topics. In the next section, in addition to other

[2] www.wired.co.uk/article/elon-musk-artificial-intelligence-world-war-3

features used in related work, we introduce our approach that leverages the author's expertise that we infer from the biography and the textual content the author has posted. Note that the aim is not to propose a novel method that infers expertise but rather to study its contribution to rank news feed updates.

Fig. 6.1: Tweet by *Elon Musk* on Artificial Intelligence (AI)

6.3 Proposed approach

The proposed approach takes as input a set of tweets $\boldsymbol{F(u)}$ unread by the recipient user $\boldsymbol{u}$ that can be included in the news feed, to predict and output a relevance score to each tweet $\boldsymbol{t \in F(u)}$. This approach is based on the personalized model we introduced in the chapter 4 and uses supervised *random forest* classifiers as relevance prediction models [43], which we found to be the most effective following the comparative study in the chapter 5. In addition to the features used in related work, our approach leverages the author's expertise that we infer from the biography and tweets the author has posted.

Let denote by $\boldsymbol{S}$ the subset of recipient users for whom we apply the proposed approach and $\boldsymbol{D(u)}$ a subset of tweets previously read by a user $\boldsymbol{u \in S}$. In order to train supervised and personalized prediction models based on *random forest*, we first create a training database for each recipient user $\boldsymbol{u \in S}$. The training database is a set of input-output pairs, called *instances*, built considering each previously read tweet $\boldsymbol{t' \in D(u)}$. An input represents a vector of features that may influence the relevance of a tweet $\boldsymbol{t' \in D(u)}$ to the user $\boldsymbol{u}$, and the corresponding output represents the implicit relevance score $\boldsymbol{R(t',u)}$ that measures the relevance of $\boldsymbol{t'}$ to $\boldsymbol{u}$. The proposed

approach involves three steps: (1) assign implicit relevance scores to tweets; (2) extract the features that may influence the relevance of tweets; and (3) train the relevance prediction model. In this section, we describe each of the steps.

6.3.1 Relevance scores

The implicit method we use assumes that a previously read tweet $\boldsymbol{t'} \in \boldsymbol{D(u)}$ is relevant to a user $\boldsymbol{u}$ if $\boldsymbol{u}$ interacted with $\boldsymbol{t'}$. Predicting relevance scores results in a binary classification problem:

$$\boldsymbol{R(t',u)} = \begin{cases} 1 & \text{if } \boldsymbol{u} \text{ interacted with } \boldsymbol{t'} \text{ (retweet or reply or like)} \\ 0 & \text{otherwise} \end{cases} \tag{6.1}$$

6.3.2 Features that may influence relevance

We extract, preprocess, and create from scratch 16 features that may influence the relevance score $\boldsymbol{R(t,u)}$ that measures the relevance of a tweet $\boldsymbol{t}$, posted by an author $\boldsymbol{u'}$, to the recipient $\boldsymbol{u}$. These features are summarized in Table 6.1 and divided into five categories. Four categories of features were used in related work and were previously presented in section 4.2.2, while we propose the features *f6*, *f7*, and *f8* to compute the fifth category corresponding to expertise. The features 1, 2, 4, 5, 6, 7, and 10 are gradually recalculated and updated as new tweets are injected into the news feed from least recent to most recent. In the rest of the section, we provide a detailed description of the features *f6*, *f7*, and *f8* that we use to compute expertise.

Publishing rate of $\boldsymbol{u'}$ for keywords of $\boldsymbol{t}$ Xu et al. [81] assert that a user is likely to have expertise in topics on which he frequently expresses his opinion. As we represent the topics of the tweets by keywords extracted using *DBpedia Spotlight*, we propose to calculate this feature using Equation 6.2. The assumption is that the author $\boldsymbol{u'}$ is an expert in the topics of $\boldsymbol{t}$ if he frequently posts tweets about these topics. We recall that our aim is not to propose a novel method that infers expertise but rather to study its contribution to rank news feed updates.

$$f_6(\boldsymbol{u'},\boldsymbol{t}) = \frac{\sum_{i=1}^{nbk(\boldsymbol{t})} Post(\boldsymbol{u'}, k_i(\boldsymbol{t}))}{nbk(\boldsymbol{t}) \times nbp(\boldsymbol{u'})} \tag{6.2}$$

Where:

Table 6.1: Features that may influence relevance

Features that may influence relevance		**Type**	**N°**
Relevance of the content of ***t***, its hashtags, and mentions to ***u***	Relevance of the keywords of ***t*** to ***u***	Int	*f1*
	Relevance of the hashtags of ***t*** to ***u***	Int	*f2*
	Presence of ***u*** in the mentions of ***t***	Bool	*f3*
Social tie strength between ***u*** and ***u'***	Interaction rate of ***u*** with tweets of ***u'***	Float	*f4*
	Number of times ***u*** mentioned ***u'***	Int	*f5*
Expertise of *u'* in the topics of *t*	**Publishing rate of *u'* for keywords of *t***	**Float**	***f6***
	Interaction rate with keywords of *t* posted by *u'*	**Float**	***f7***
	Keywords of *u'* biography and keywords of *t*	**Bool**	***f8***
Authority of ***u'***	Followers count / Followings count	Int	*f9*
	Seniority in years	Int	*f10*
	Listed (group) count	Int	*f11*
Quality of ***t***	Length (# characters)	Int	*f12*
	Presence of hashtags	Bool	*f13*
	Presence of a URL	Bool	*f14*
	Presence of a photo or a video	Bool	*f15*
	Popularity (# retweets, replies, likes)	Int	*f16*

- $k_i(\boldsymbol{t})$ is the i^{th} keyword of $\boldsymbol{t}$
- $nbk(\boldsymbol{t})$ is the number of keywords of $\boldsymbol{t}$
- $nbp(\boldsymbol{u'})$ is the number of tweets previously posted by $\boldsymbol{u'}$
- $Post(\boldsymbol{u'}, k_i(\boldsymbol{t}))$ is the number of times $\boldsymbol{u'}$ has previously posted $k_i(\boldsymbol{t})$

For example, if $\boldsymbol{t}$ has 3 keywords k_1, k_2, k_4 and $\boldsymbol{u'}$ has previously posted 2 tweets t_1 and t_2, that have respectively the keywords k_1, k_2 and k_1, k_3, then the following value will be assigned to this feature:

$$f_6(\boldsymbol{u'},\boldsymbol{t}) = \frac{2+1+0}{3\times 2} = 0.5 \tag{6.3}$$

Interaction rate with keywords of $\boldsymbol{t}$ posted by $\boldsymbol{u'}$ Li et al. [106] state that tweets on specific topics authored by a recognized expert in his field may attract the attention of users and get more interactions. Indeed, as shown in the example in Fig. 6.1, the more knowledgeable a user is about the topic on which he has posted a tweet, the more relevant and engaging the tweet is for users. Unlike the biography, which may contain self-reported expertise indicators, the user interactions reflect external expertise indications of a given user, i.e. the judgment of a user's expertise by other users. We propose to calculate this feature using Equation 6.4. The assumption is that the author $\boldsymbol{u'}$ is an expert in the topics of $\boldsymbol{t}$ if the followers of $\boldsymbol{u'}$ have often interacted (retweet, reply, like) with tweets that $\boldsymbol{u'}$ posted on the same topics.

$$f_7(\boldsymbol{u'},\boldsymbol{t}) = \frac{\sum_{i=1}^{nbk(\boldsymbol{t})} Interaction(\boldsymbol{u'}, k_i(\boldsymbol{t}))}{\sum_{i=1}^{nbk(\boldsymbol{t})} Post(\boldsymbol{u'}, k_i(\boldsymbol{t})) \times nbf(\boldsymbol{u'}) \times 3} \tag{6.4}$$

Where:

- $k_i(\boldsymbol{t})$ is the i^{th} keyword of $\boldsymbol{t}$
- $nbk(\boldsymbol{t})$ is the number of keywords of $\boldsymbol{t}$
- $Post(\boldsymbol{u'}, k_i(\boldsymbol{t}))$ is the number of times $\boldsymbol{u'}$ has previously posted $k_i(\boldsymbol{t})$
- $nbf(\boldsymbol{u'})$ is the number of followers of $\boldsymbol{u'}$
- 3 is the maximum number of interactions a user can perform on $\boldsymbol{t}$, which corresponds to the case where the same user retweet, like, and reply to $\boldsymbol{t}$

- *Interaction*($\boldsymbol{u'}$, $k_i(\boldsymbol{t})$) is the number of interactions with tweets previously posted by $\boldsymbol{u'}$ that have the keyword $k_i(\boldsymbol{t})$

For example, if $\boldsymbol{t}$ has 3 keywords k_1, k_2, k_4, and $\boldsymbol{u'}$ has 20 followers and has previously posted 2 tweets t_1 and t_2, that have respectively the keywords k_1, k_2 and k_1, k_3, and t_1 and t_2 have get respectively 35 and 15 interactions, then the following value will be assigned to this feature:

$$f_7(\boldsymbol{u'},\boldsymbol{t}) = \frac{50+35+0}{(2+1+0)\times 20\times 3} = 0.47 \tag{6.5}$$

Keywords of $\boldsymbol{u'}$ biography and keywords of $\boldsymbol{t}$ According to Wagner et al. [78], self-reported user biographies often contain information that indicates the expertise of users such as their education, skills, career information, etc. We propose to calculate this feature using Equation 6.6. The assumption is that the author $\boldsymbol{u'}$ has expertise in the topics of $\boldsymbol{t}$ if the biography of $\boldsymbol{u'}$ includes keywords from the tweet $\boldsymbol{t}$. For example, if a user indicates in the biography that he graduated in Artificial Intelligence (AI) and the user posts a tweet about AI, then the user is likely to have expertise in that tweet.

$$f_8(\boldsymbol{u'},\boldsymbol{t}) = \begin{cases} 1 & \text{if } |kb(\boldsymbol{u'}) \cap k(\boldsymbol{t})| > 0 \\ 0 & \text{otherwise} \end{cases} \tag{6.6}$$

Where:

- $k(\boldsymbol{t})$ is the set of keywords of $\boldsymbol{t}$
- $kb(\boldsymbol{u'})$ is the set of keywords of the biography of $\boldsymbol{u'}$, extracted in the same way as the keywords of tweets

6.3.3 Relevance prediction model

First, considering each previously read tweet $\boldsymbol{t' \in D(u)}$ from least recent to most recent, we create the training database instances of each recipient user $\boldsymbol{u \in S}$ in the form of input-output pairs. An input represents a vector of features that may influence the relevance of $\boldsymbol{t'}$ to $\boldsymbol{u}$, and the output represents the implicit relevance score $\boldsymbol{R(t',u)}$.

Second, we divide the training database of each recipient user $\boldsymbol{u \in S}$ into two sets: a training set of the relevance prediction model for 70% of the first instances (the least recent ones) and a test set for 30% of the remaining instances (the most recent

ones). The latter set will be used to evaluate the performance of the prediction model.

Finally, to create a personalized prediction model for each user $\boldsymbol{u} \in \boldsymbol{S}$, we use the corresponding training set to train a supervised *random forest* classifier [43], which fits several *decision tree* estimators [41]. The purpose is to use multiple *decision trees* learned from previously read tweets in the training set to map new input features of a tweet unread by the user $\boldsymbol{u}$ to a relevance score. The *random forest* algorithm has been extremely successful as a classification and regression method in a wide range of prediction problems such as data science, bioinformatics, 3D object recognition, etc. [99]. In the chapter 5, we compared several machine learning algorithms used in related work. We found that ensemble learning models such as *Gradient Boosting* and *Random Forest* are the most suitable to predict the relevance of news feed updates.

Random Forest is an *ensemble learning* method[3] for classification and regression problems that operate by constructing a multitude of *decision trees* [89]. Each tree is distinguished by the sub-sample vector on which it is trained, and which is randomly selected with the same distribution from the training set [43]. This method uses averaging to improve the predictive performance and correct the decision trees' habit of overfitting [89]. It outputs the class that is the mode of the classes in case of classification and the mean prediction of the individual trees in case of regression [89]. As discussed in section 6.3.1, we recall that predicting relevance scores of tweets is a binary classification problem. The rest of the section therefore focuses on *decision tree* and *random forest* classifiers. In this case study, as shown in Fig. 6.2, a *decision tree* classifier for a recipient user $\boldsymbol{u} \in \boldsymbol{S}$ is a flowchart-like structure in which a node represents a test on a feature that may influence relevance, a branch represents the outcome of the test, a leaf node represents the class label and relevance score of a tweet $\boldsymbol{t}$, and a path from the root to a leaf node corresponds to a relevance classification rule. For example in Fig. 6.2, an illustration of a classification rule from the *decision tree* of the *Twitter* user *Medium* is as follows: if the keywords of the tweet $\boldsymbol{t}$ are irrelevant to the recipient $\boldsymbol{u}$, and the interaction rate of $\boldsymbol{u}$ with tweets posted by the author $\boldsymbol{u'}$ is lower than 50%, then the tweet $\boldsymbol{t}$ is irrelevant to $\boldsymbol{u}$. Note that more details about *decision trees* are provided in [41]. Regarding *random forest* classifiers, Breiman [43] proposes the following definition:

[3] A method that uses multiple machine learning algorithms to obtain better predictive performance than could be obtained from any of the constituent learning algorithms alone.

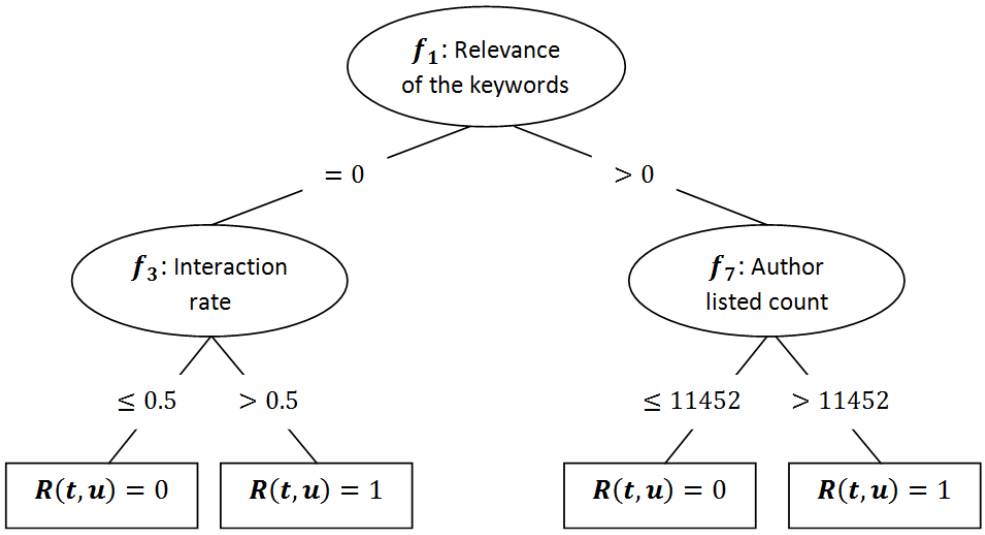

Fig. 6.2: *Decision tree* for the *Twitter* user *Medium*

Definition. A *random forest* classifier consists of a collection of tree-based classifiers $\{h(\mathbf{x}\ , \Theta_k),\ k = 1, ...\}$ where the $\{\Theta_k\}$ are independent identically distributed random vectors. Each tree in the forest casts a unit vote for the most popular class at input $\mathbf{x}$.

According to the previous definition, as reported by Malekipirbazari and Aksakalli [107], and as shown in Fig. 6.3, the *random forest* classifier methodology can be formally described in the following three steps:

1. a random sub-sample vector Θ_k is created from the training set for the k^{th} decision tree. The independence property enforces that the random vector Θ_k is independent of the past random vectors $\Theta_{1\ ...}\ \Theta_{k\text{-}1}$, but with the same distribution;

2. multiple *decision tree* classifiers are build so that the k^{th} tree is build using the random vector Θ_k, resulting in a classifier $h(\mathbf{x}\ , \Theta_k)$ where $\mathbf{x}$ is an input feature vector.

3. to classify a new tweet from an input feature vector $\mathbf{x}$, the tweet is presented to each of the trees in the forest. Each tree first gives a classification output, i.e. relevant tweet or not, then the forest chooses the classification having the most votes. To combine *decision tree* classifiers $\{h_1(\mathbf{x}),\ h_2(\mathbf{x}),\ ...\ ,\ h_k(\mathbf{x})\}$ using the training set drawn randomly from the distribution of the random vector $\mathbf{X}$, Y, the margin function mg can be defined with the following equation:

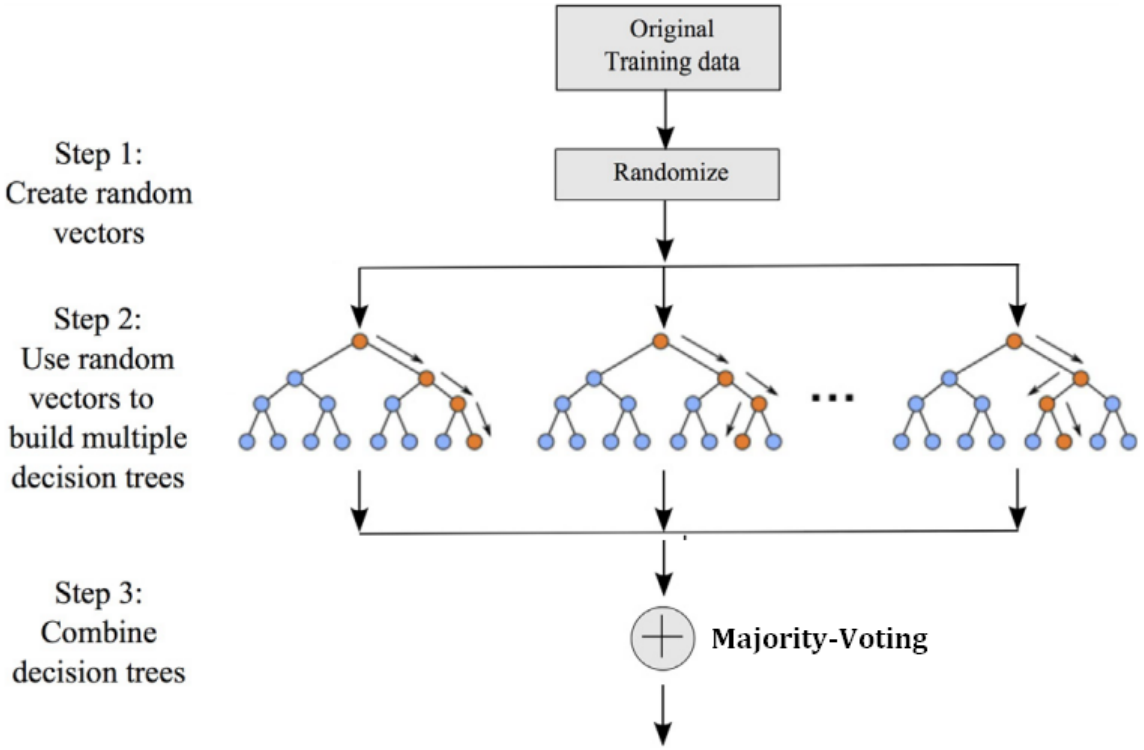

Fig. 6.3: The *random forest* methodology [107]

$$mg(\mathbf{X}, Y) = av_k I(h_k(\mathbf{X}) = Y) - \max_{j \neq Y} av_k I(h_k(\mathbf{X}) = j) \tag{6.7}$$

Where av is the average and I the indicator function [108]. The margin measures the extent to which the average number of votes at $\mathbf{X}$, Y for the right class exceeds the average vote for any other class. Examples on the construction of *random forests* are available in [89].

6.4 Experiments and results

To evaluate the performance of the proposed approach and investigate the contribution of expertise to rank news feed updates, we describe in this section the measures used to evaluate the performance and the obtained evaluation results.

6.4.1 Measures

In the experiments, we used the dataset described in section 4.5.1 and the concepts TP, TN, FP, and FN defined in section 4.5.2. First, as discussed in section 6.3.3, we train a personalized *random forest* classifier for each recipient user $\boldsymbol{u} \in \boldsymbol{S}$ using the corresponding training set, which represents 70% of the least recent instances in the training database. Then, we evaluate model performance using the corresponding test set (30% of the most recent instances) and study the contribution of expertise to rank news feed updates. To do so, we use the weighted *F1 score* measure denoted by F and given by Equation 6.8 [91].

$$F = \frac{(F_r \times (TP + FN)) + (F_i \times (TN + FP))}{TP + TN + FP + FN} \tag{6.8}$$

Where:

- F_r is the *F1 score* for the relevant tweets class
- F_i is the *F1 score* for the irrelevant tweets class

Finally, to investigate the impact of expertise, we use the F *score* and the corresponding test set of each user $\boldsymbol{u} \in \boldsymbol{S}$ to perform experiments and compare two approaches: our approach that uses the author's topical expertise and a classical approach that is the same as ours except that it does not use it. The contribution of expertise for a user is denoted by C and given by Equation 6.9. Moreover, to investigate the importance of the author's expertise compared to other features used in related work, we compute the importance scores of features when judging the relevance of tweets by recipient users. As described by Breiman and Cutler [89], the importance of a feature is calculated as the normalized total reduction of the criterion brought by that feature and is known as the *Gini importance*. The higher the value, the more important the feature. More details about the importance of features are provided in [89].

$$C = F \text{ with expertise} - F \text{ without expertise} \tag{6.9}$$

6.4.2 Results

In the experiments, we first selected the best *random forest* parameters (number of trees, maximum three depth, splitting criterion, etc.) with a *random search* [92] and a 5-fold time-series cross-validation performed on the train set [91]. A time-series

cross-validation was used instead of the standard cross-validation to preserve the chronological order of tweets. To find the best parameters of both expertise and non-expertise-aware models, a *random search* was hence run over different parameter values. Finally, to study the algorithmic stability with several runs and small changes to training data, we retrained and iterated both prediction models on 30 different *random state*[4] values, then evaluated the models on the test set. The average *F score* was selected for both expertise and non-expertise-aware approaches.

First, Table 6.2 presents the average experimental results of the comparison between the proposed expertise-aware approach and the classical approach, and Fig. 6.4 summarizes the contribution of the author's expertise for all recipient users. The results of Table 6.2 show that our approach most often succeeds in predicting relevance scores of tweets with an average *F score* of 81.59%. Furthermore, we point out that the proposed approach gives remarkable results for several recipient users with an *F score* of more than 90%: 100% for *TheMuslimReform*, 97.89% for *john_walsh*, 96.67% for *SfNtweets*, etc. Certainly, this proves that the features and *random forest* models we used are overall very effective to predict the relevance on the one hand, and on the other hand, they are perfectly adapted to rank news feed updates for several users. The results of Table 6.2 also show that our expertise-aware approach has overall improved the classical approach by an average gain of +%2,71, from an average *F score* of 78,88% to %81,59. Undoubtedly, this highlights that judging expertise, which has not been considered in the academic and the industrial communities, is critical for recommending relevant updates on particular topics. The results of Table 6.2 and Fig. 6.4 indicate that the gain brought by the author's expertise is positive for 63% of recipient users when predicting the relevance of tweets. This suggests that the inclusion of expertise enables to refine the prediction and maximize the relevance in news feeds for more than three users out of five, which is significant. Moreover, we notice from Table 6.2 that infer the author's expertise made a high contribution to several users, e.g. +25.36% for *GilmoreGuysShow*, +14.03% for *LKrauss1*, +10.3% for *NASA_Wallops*, etc. Indeed, this confirms that infer expertise enables recipient users, especially users who value the author's proficiency, to catch up with the valuable and reliable content on specific topics.

[4]A variable used in randomized algorithms to determine the random seed of the pseudo-random number generator.

Table 6.2: Experimental results

User	Number of instances	F with expertise	F without expertise	C
Astro0Glen	1245	83,26	81,61	**1,65**
Astro__Pam	837	72,83	73,77	**-0,94**
bamwxcom	123	86,62	86,62	**0**
Baronatrix	220	76,43	77,79	**-1,36**
BethStamper6	760	72,57	71,93	**0,64**
byudkowsky	185	70,51	70,51	**0**
ch402	497	86,12	79,66	**6,46**
demishassabis	319	86,58	88,5	**-1,92**
eevil_abby	148	77,41	78,94	**-1,53**
elonmusk	303	91,13	92,34	**-1,21**
GeorgeHarrison	177	85,49	79,76	**5,73**
GilmoreGuysShow	105	90,2	64,84	**25,36**
gwern	771	69,54	70,26	**-0,72**
homebrew	1115	81,89	80,21	**1,68**
HybridZizi	1039	73,87	73,31	**0,56**
jadelgador	1401	84,46	81,82	**2,64**
JHUBME	516	90,45	89,18	**1,27**
JohnDawsonFox26	1649	79,69	80,04	**-0,35**
john_walsh	154	97,89	93,73	**4,16**
kilcherfrontier	708	85,77	81,69	**4,08**
LKrauss1	115	68,32	54,29	**14,03**
mastenspace	135	90,38	80,76	**9,62**
Medium	56	87,55	80,47	**7,08**
microphilosophy	112	65,04	67,3	**-2,26**
MIRIBerkeley	195	87,87	87,87	**0**
NASAKepler	86	83,23	74,84	**8,39**
NASA_Wallops	116	80,99	70,69	**10,3**
newscientist	281	78,66	75,74	**2,92**
PattiPiatt	1173	80,05	80,11	**-0,06**
peterboghossian	1388	74,83	69,96	**4,87**
rafat	782	81,87	81,53	**0,34**
realDonaldTrump	140	85,89	88,17	**-2,28**
Red_or_MC1R	306	72,09	69,26	**2,83**
renormalized	543	73,77	74,9	**-1,13**
RossTuckerNFL	772	89,21	87,11	**2,1**
RoxanneDawn	366	83,24	84,37	**-1,13**
scimichael	2940	79,87	78,45	**1,42**
SfNtweets	198	96,67	96,67	**0**
slatestarcodex	271	65,85	64,77	**1,08**
SLSingh	560	73,34	68,26	**5,08**
sxbegle	382	79,25	77,07	**2,18**
TeslaRoadTrip	1692	86,71	84,34	**2,37**
TheMuslimReform	27	100	100	**0**
TheRickDore	202	86,51	82,81	**3,7**
USDISA	522	76,19	71,48	**4,71**
WestWingWeekly	548	82,83	80,7	**2,13**
Average	**569**	**81,59**	**78,88**	**2,71**

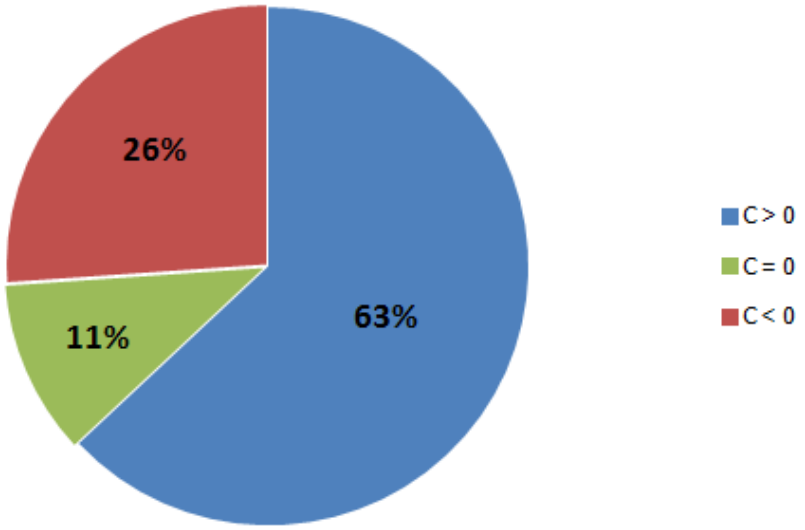

Fig. 6.4: The contribution of expertise (C) for all users

Second, we computed the average feature importance scores for all recipient users, which are presented in Fig. 6.5. The results show that all the features we used are important in judging the relevance of tweets. The results show that the top feature is the feature f_{16} (0.18) which measures the tweet's popularity. Indeed, the more users interact with a tweet, the more likely it is of high quality [4]. The second most important feature is the feature f_4 (0.12), the interaction rate of $\boldsymbol{u}$ with tweets posted by $\boldsymbol{u'}$. Certainly, if $\boldsymbol{u}$ found tweets posted by $\boldsymbol{u'}$ relevant in the past, then $\boldsymbol{u}$ may find the tweets of $\boldsymbol{u'}$ relevant in the future [65]. The third most important feature is the feature f_{12} (0.11) which measures the tweet length. Indeed, a long tweet is likely to be more formal, more informative, and of better quality [64] . The results also indicate that the features f_6 and f_7 (0.07 and 0.09 respectively) which we introduced to infer the author's expertise from the tweets posted are extremely important. Undoubtedly, tweets posted by a user have been used in several works to infer expertise [80], and leveraging this expertise appears to be very effective to predict and judge the relevance of tweets. Moreover, the results show that the features f_9, f_{10}, and f_{11} (0.08, 0.06, and 0.1 respectively) which measure the author's authority are very important. Indeed, if a user is important, then his updates are also important [69]. We further notice that the features f_1, f_2, and f_3 (0.05, 0.02, and 0.04 respectively) which measure respectively the relevance of the content of $\boldsymbol{t}$, its hashtags, and mentions to $\boldsymbol{u}$, are not the most important. This proves that predicting relevance scores is a difficult task because the

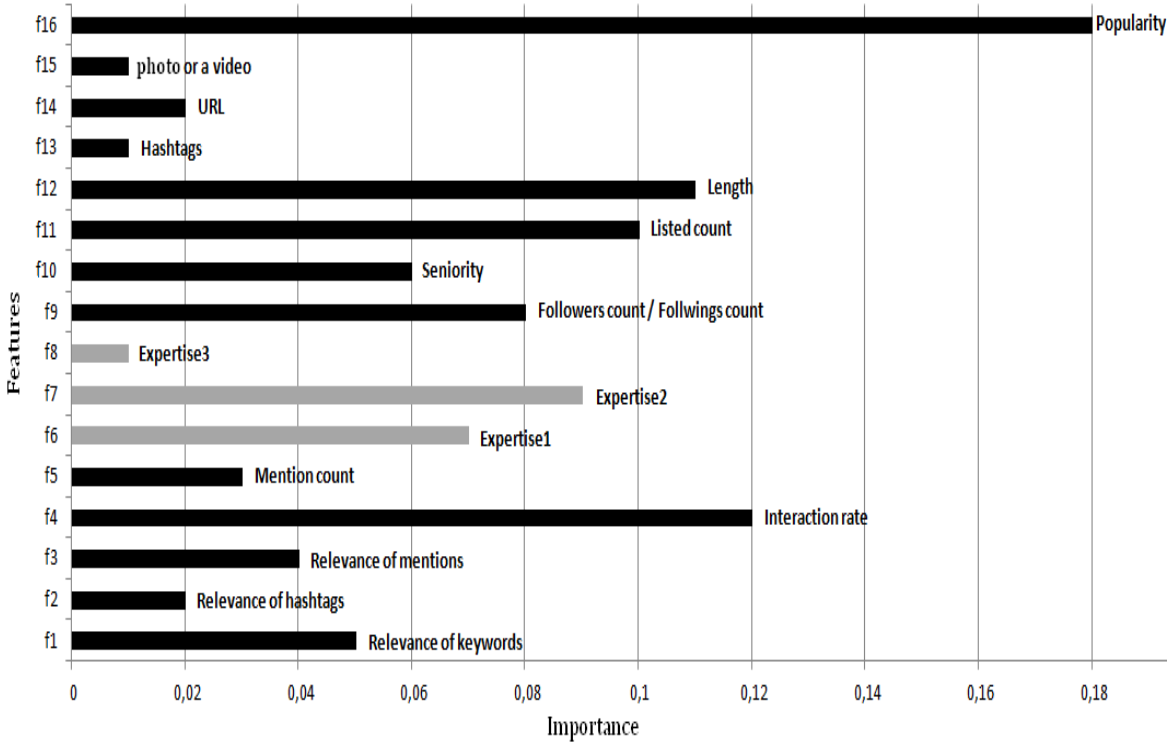

Fig. 6.5: Average feature importance for users

most important features are not the most intuitive. Furthermore, we note that the features f_5 (0.03) which represents the number of times ***u*** mentioned ***u'*** in tweets he posted is not very important. Indeed, it is not common for users to mention other users, except for dialogues [55]. The results also reveal that the feature f_8 (0.01) which we used to infer the author's expertise from the biography is surprisingly not important. This is probably due to the fact that users do not frequently change their biography and that this feature is sparse since not all users provide a full description [105]. Finally, we notice that the features f_{13}, f_{14}, and f_{15} (0.01, 0.02, and 0.01 respectively) which measure tweet quality are not really important since they are non-personalized features that do not take into consideration the preferences of individual users [65].

Despite the improvements we have made, we observe that the proposed approach has limitations for a small number of users. First, from Table 6.2, the experimental results show that our approach gives modest results for a few users with an *F score* of less than 70%: 69.54% for *gwern*, 68.32% for *LKrauss1*, 65.85% for *slatestarcodex*, and 65.04% for *microphilosophy*. The number of instances in the training database does not seem to be the main cause. Indeed, a higher number of instances does not necessarily

involves better results and vice-versa, e.g. an *F score* of %69,54 for *gwern* with 771 instances comparing to an *F score* of %87,55 for *Medium* with only 56 instances. Therefore, we believe that these modest results are mainly due to: (1) the fact that we do not have the browsing information of users, which is crucial to our approach. More specifically, we are not sure whether a recipient user has read a tweet or not in case of non-interaction. This certainly affects the results for both training and evaluation; and (2) the implicit training and evaluation method and more precisely the FP measure, which was on average 13 irrelevant tweets incorrectly predicted relevant, may wrongly penalize our approach. Indeed, non-interaction does not always mean irrelevance. A user can find a tweet relevant and choose not to interact with it, or simply miss it.

Second, from Table 6.2 and Fig. 6.4, the results show that infer the author's expertise has no contribution to 11% of recipient users when predicting relevance scores of tweets, e.g. *byudkowsky* and *SfNtweets*. The results also indicate that the gain brought by this feature is negative for 26% of users, e.g. -1.21% for *elonmusk*, -1.53% for *eevil_abby*, and -1.13% for *renormalized*. We believe that these results are due to: (1) other features that we did not use to infer the author's expertise but might be important for these recipient users when judging the relevance of tweets, e.g. the author's list memberships, the social relationships, etc. [80]; (2) the fact that *Twitter* is a general public social media and not a specialized, professional, or academic one. Indeed, different types of tweets are encountered by users in their news feed. Some tweets are related to highly specialized areas where expertise is important, e.g. technology, science, sports, etc., and some tweets are not, e.g. tweets about friends, routine activities, personal experiences, etc. [86]; and (3) user information needs, which are different on social media [79]. Certainly, a minority of recipient users may not give importance to the author's expertise when judging the relevance of tweets.

6.5 Conclusion

In this chapter, we first analyzed and discussed research work in the field of expert finding on social media. Then, to investigate the contribution of expertise to rank news feed updates, we proposed a personalized approach that predicts the relevance using supervised prediction models based on *random forest*. The proposed approach uses the author's expertise that we inferred from the biography and the textual content the author has posted, in addition to the four types of features used in related work:

(1) the relevance of the update content to the recipient's interests; (2) the social tie strength between the recipient and the author; (3) the author's authority; and (4) the update quality. The experimental results on *Twitter* show that the features and *random forest* models we used overall succeed in predicting the relevance of tweets with remarkable results for several recipient users. Moreover, the results show that infer and introduce the author's expertise has improved the classical approach with a considerable gain in prediction for many users. The feature importance also showed that expertise is ranked among the top features when judging the relevance of tweets by recipients users. All these findings indicate that leverage the author's expertise is critical for recommending valuable updates and maximizing the relevance in news feeds. However, given the limitations we identified for some users, efforts must still be made to improve the proposed approach.

Chapter 7

Conclusion and future work

7.1 Conclusion

In this thesis, we addressed the problem of ranking news feed updates on social media to provide recommendations and help users quickly find relevant updates from the large and overwhelming stream of data. The aim was to propose machine-learning-based approaches to rank and display news feed updates in descending relevance order based on the prediction of a relevance score between a user and a new update unread in the news feed. To address this problem, we have made several contributions.

In the first part of the state-of-the-art, we first identified the main challenges on social media due to the increasing amount of data. Then, we provided background on recommender systems and described three main classical recommendation approaches: content-based filtering, collaborative filtering, and hybrid filtering. Following this study, it turns out that each approach has advantages but also disadvantages. Therefore, the choice of one recommendation approach is strongly linked to the problem being addressed and the data available to the recommendation system. Finally, we introduced the machine learning aspect, which is becoming increasingly important in the recommendation, as it overcomes the limitations of classical recommendation systems that fail to cope with the increasing data generated by social media and the unique specificity of their social structural information. In the machine learning aspect, we discussed the two main approaches used in the learning process: supervised learning such as classification and regression, and unsupervised learning such as clustering and dimensionality reduction.

In the second part of the state-of-the-art, we first provided background on ranking news feed updates on social media, including statistics on data volume and irrelevance that confirm the need for ranking. Then, we discussed research works carried out in the field of ranking news feed updates and exposed their advantages and limitations according to four main criteria: (1) the features that may influence relevance; (2) the relevance prediction models; (3) the training and evaluation methods; and (4) the evaluation platforms. Finally, we identified several open research issues. Following the state-of-the-art, it appeared that research in the field of ranking news feed updates is not completely achieved. Indeed, several approaches have been proposed, implemented, and evaluated. However, given the limitations we identified, we found that efforts should be made to improve the ranking process.

In the first contribution, we first provided a reminder of the classical non-personalized approach used in related work to rank news feed updates, which uses a prediction model for all users. Then, to predict the relevance of news feed updates and improve user experience, we used the *random forest* algorithm to train and introduce a personalized prediction model for each user. Finally, we conducted a comparative study of personalized and non-personalized models according to several criteria. The experimental results on a dataset crawled from *Twitter* showed that a single non-personalized model for all users is easy to manage and fine-tune, is less likely to overfit as it benefits from more training data, and it addresses the problem of cold-start and inactive users, which is a common problem in recommender systems. On the other hand, the personalized models we introduce allow personalized feature importance for users, take into consideration the preferences of each user, and allow to track changes in user preferences over time. Furthermore, the personalized models we propose give a higher prediction accuracy than non-personalized models. These findings highlight the need for personalization to effectively rank the news feed, promote relevant updates, and assist users by suggesting tailored content of interest.

In the second contribution, we first described the context of the comparison according to a personalized approach that predicts the relevance of news feed updates. Then, we selected and described seven supervised algorithms that have been used in related work to predict the relevance. After that, we defined a rigorous and fair comparison methodology by selecting the best parameters of each algorithm according to the data. Finally, to determine the most suitable models, we conducted a comparative

study by evaluating, analyzing, and comparing the selected algorithms according to several criteria. Following extensive experiments on *Twitter*, the results highlight that selecting the most suitable supervised model is critical to increase the efficiency of the ranking process. Furthermore, the results showed that ensemble learning models such as *Gradient Boosting* and *Random Forest* are the most appropriate to predict the relevance of updates. Indeed, the comparison results showed that these models are fast in both training and predicting, that they are scalable, and they outperform the other algorithms in prediction on various training set sizes. Moreover, they become very accurate as the data size increases.

In the third contribution, we first analyzed and discussed research work in the field of expert finding on social media. Then, to investigate the contribution of expertise to rank news feed updates, we proposed a personalized approach that predicts the relevance using supervised prediction models based on *random forest*. In addition to other types of features used in related work, the proposed approach leverages the author's expertise that we inferred from the biography and the textual content the author has posted. The experimental results on *Twitter* showed that the features and *random forest* models we used overall succeed in predicting the relevance of updates with remarkable results for several recipient users. Moreover, the results show that infer and introduce the author's expertise has improved the classical approach with a considerable gain in prediction for many users. The feature importance also showed that expertise is ranked among the top features when judging the relevance of updates by recipients users. All these findings indicate that leverage the author's expertise is critical to recommend valuable updates and maximize the relevance in news feeds.

7.2 Future work

The contributions we have made to predict the relevance and rank news feed updates can be further improved in several ways. Indeed, some of the most promising research directions and improvements are as follows:

- For now, like most related work, we only evaluated our approaches using the implicit training and evaluation method, which assumes that a user's interaction (retweet, reply, like) with an update involves its relevance. However, non-interaction does not always mean irrelevance. For example, a user can find an update relevant and choose not to interact with it. For further work, we

intend to get explicit user feedback by asking their opinion on the predicted relevance scores. Furthermore, it would be interesting to study and understand the relationship between the implicit and the explicit method. For example, investigate the correlation between user interactions and their explicit relevance ratings to the news feed updates. Finally, in case of non-interaction, it would be interesting to include the reading time as an implicit indicator of relevance, based on the assumption that the more time a user spends reading an update, the more likely the update is relevant to that user.

- Despite the high predictive accuracy obtained with the predictive features we used, we believe that the proposed approaches can still be extended with other features that may influence the relevance and that we did not use in this work. For example, it would be interesting to integrate the freshness of news feed updates in terms of date and time. Furthermore, concerning the expertise we have introduced to predict the relevance, it would be worthwhile to include other features that we did not use to infer the author's expertise but might be important for recipient users when judging the relevance of updates, e.g. the author's list memberships, the social relationships, etc. Finally, it would be interesting to conduct a study to identify the characteristics of recipient users who value expertise.

- Despite the many advantages that personalized models we introduced have brought over the classical non-personalized models, we observed that non-personalized models may still work better with new or inactive users, for which personalized models may have very few training instances. Indeed, in such cases, the personalized model does not have information about user preferences and interests to make specific recommendations. Hence, it is important to suggest alternatives to address this common problem in recommender systems known as the *cold-start problem*. Non-personalized models address this issue by default since the same model can be used for any user, even new or inactive users. To address this problem, it would be interesting to propose a hybrid method that takes the advantages of both personalized and non-personalized models.

- Despite the high performance, the speed, and the scalability of the random forest algorithm we used, this algorithm remains a shallow learning model whose performance depends partly on the manually extracted features. *Artificial neural network* models have been used in related work to process the textual content so

that the embeddings that have been used encode only information about the words of the textual content of the updates. However, we have seen in this work that other features such as those measuring the author's authority and the social tie strength between the author and the recipient are extremely important to predict the relevance. To the best of our knowledge, *deep neural network* models that automatically execute feature extraction have not yet been used to predict the relevance of news feed updates. Therefore, for further improvement, it would be interesting to study the feasibility of deep learning models to automatically extract all the features that may influence relevance and compare their performance with ensemble methods such as *Gradient Boosting* and Random Forest.

References

[1] Shlomo Berkovsky and Jill Freyne. Personalised Network Activity Feeds: Finding Needles in the Haystacks. In Martin Atzmueller, Alvin Chin, Christoph Scholz, and Christoph Trattner, editors, *Mining, Modeling, and Recommending 'Things' in Social Media*, volume 8940, pages 21–34. Springer International Publishing, Cham, 2015.

[2] Shlomo Berkovsky, Jill Freyne, and Gregory Smith. Personalized Network Updates: Increasing Social Interactions and Contributions in Social Networks. In David Hutchison, Takeo Kanade, Josef Kittler, Jon M. Kleinberg, Friedemann Mattern, John C. Mitchell, Moni Naor, Oscar Nierstrasz, C. Pandu Rangan, Bernhard Steffen, Madhu Sudan, Demetri Terzopoulos, Doug Tygar, Moshe Y. Vardi, Gerhard Weikum, Judith Masthoff, Bamshad Mobasher, Michel C. Desmarais, and Roger Nkambou, editors, *User Modeling, Adaptation, and Personalization*, volume 7379, pages 1–13. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012.

[3] Pritam Gundecha and Huan Liu. Mining social media: a brief introduction. *Tutorials in Operations Research*, 1(4), 2012.

[4] Li Kuang, Xiang Tang, MeiQi Yu, Yujian Huang, and Kehua Guo. A comprehensive ranking model for tweets big data in online social network. *EURASIP Journal on Wireless Communications and Networking*, 2016(1):46, December 2016.

[5] Keyi Shen, Jianmin Wu, Ya Zhang, Yiping Han, Xiaokang Yang, Li Song, and Xiao Gu. Reorder user's tweets. *ACM Transactions on Intelligent Systems and Technology*, 4(1):1–17, January 2013.

[6] Azin Ghazimatin, Rishiraj Saha Roy, and Gerhard Weikum. FAIRY: A Framework for Understanding Relationships Between Users' Actions and their Social Feeds. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, pages 240–248. ACM, 2019.

[7] Kaisong Song, Daling Wang, Shi Feng, Yifei Zhang, Wen Qu, and Ge Yu. CTROF: A Collaborative Tweet Ranking Framework for Online Personalized Recommendation. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 1–12. Springer, 2014.

[8] Deepak Agarwal, Bee-Chung Chen, Rupesh Gupta, Joshua Hartman, Qi He, Anand Iyer, Sumanth Kolar, Yiming Ma, Pannagadatta Shivaswamy, and Ajit Singh. Activity ranking in LinkedIn feed. In *Proceedings of the 20th ACM*

SIGKDD international conference on Knowledge discovery and data mining, pages 1603–1612, 2014.

[9] Deepak Agarwal, Bee-Chung Chen, Qi He, Zhenhao Hua, Guy Lebanon, Yiming Ma, Pannagadatta Shivaswamy, Hsiao-Ping Tseng, Jaewon Yang, and Liang Zhang. Personalizing linkedin feed. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1651–1660, 2015.

[10] Yinbo Zhu, Zhenyu Wang, Yiqun Wu, Zhenhua Huang, Min Li, and Rong Zeng. Tweets Ranking Considering Dynamic Social Influence and Personal Interests. In *Proceedings of the 2018 10th International Conference on Machine Learning and Computing*, pages 276–282, 2018.

[11] Magdalini Eirinaki, Jerry Gao, Iraklis Varlamis, and Konstantinos Tserpes. Recommender Systems for Large-Scale Social Networks: A review of challenges and solutions. *Future Generation Computer Systems*, 78:413–418, January 2018.

[12] MohammadNoor Injadat, Fadi Salo, and Ali Bou Nassif. Data mining techniques in social media: A survey. *Neurocomputing*, 214:654–670, November 2016.

[13] Ivens Portugal, Paulo Alencar, and Donald Cowan. The use of machine learning algorithms in recommender systems: A systematic review. *Expert Systems with Applications*, 97:205–227, May 2018.

[14] Andreas M. Kaplan and Michael Haenlein. Users of the world, unite! The challenges and opportunities of Social Media. *Business Horizons*, 53(1):59–68, January 2010.

[15] Anitha Anandhan, Liyana Shuib, Maizatul Akmar Ismail, and Ghulam Mujtaba. Social media recommender systems: review and open research issues. *IEEE Access*, 6:15608–15628, 2018. Publisher: IEEE.

[16] Luis Terán, Alvin Oti Mensah, and Arianna Estorelli. A literature review for recommender systems techniques used in microblogs. *Expert Systems with Applications*, 103:63–73, August 2018.

[17] Charu C. Aggarwal. *Recommender systems*, volume 1. Springer, 2016.

[18] J. Bobadilla, F. Ortega, A. Hernando, and A. Gutiérrez. Recommender systems survey. *Knowledge-Based Systems*, 46:109–132, July 2013.

[19] Norjihan Abdul Ghani, Suraya Hamid, Ibrahim Abaker Targio Hashem, and Ejaz Ahmed. Social media big data analytics: A survey. *Computers in Human Behavior*, 101:417–428, December 2019.

[20] Stanley Wasserman and Katherine Faust. *Social network analysis: Methods and applications*, volume 8. Cambridge university press, 1994.

[21] Martin Ester. Recommendation in social networks. In *RecSys*, pages 491–492, 2013.

[22] Pasquale Lops, Marco de Gemmis, and Giovanni Semeraro. Content-based Recommender Systems: State of the Art and Trends. In Francesco Ricci, Lior Rokach, Bracha Shapira, and Paul B. Kantor, editors, *Recommender Systems Handbook*, pages 73–105. Springer US, Boston, MA, 2011.

[23] Xiaoyuan Su and Taghi M. Khoshgoftaar. A survey of collaborative filtering techniques. *Advances in artificial intelligence*, 2009, 2009. Publisher: Hindawi.

[24] Robin Burke. Hybrid Recommender Systems: Survey and Experiments. *User Modeling and User-Adapted Interaction*, 12(4):331–370, 2002.

[25] G. Adomavicius and A. Tuzhilin. Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions. *IEEE Transactions on Knowledge and Data Engineering*, 17(6):734–749, June 2005.

[26] Thomas Hofmann and Jan Puzicha. Latent class models for collaborative filtering. In *IJCAI*, volume 99, 1999. Issue: 1999.

[27] Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix Factorization Techniques for Recommender Systems. *Computer*, 42(8):30–37, August 2009.

[28] Gene H. Golub and Christian Reinsch. Singular value decomposition and least squares solutions. In *Linear Algebra*, pages 134–151. Springer, 1971.

[29] James Bennett and Stan Lanning. The netflix prize. In *Proceedings of KDD cup and workshop*, volume 2007, page 35. New York, 2007.

[30] Christopher M. Bishop. Pattern recognition. *Machine Learning*, 128, 2006.

[31] Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar. *Foundations of machine learning*. MIT press, 2012.

[32] Jaime Guillermo Carbonell, Tom Michael Mitchell, and Ryszard Stanislaw Michalski. *Machine learning: An artificial intelligence approach*. M. Kaufmann., 1983.

[33] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. Adaptive computation and machine learning. The MIT Press, Cambridge, Massachusetts, 2016.

[34] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning*. Springer Series in Statistics. Springer New York, New York, NY, 2009.

[35] Aurélien Géron. *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow: Concepts, tools, and techniques to build intelligent systems*. O'Reilly Media, 2019.

[36] D.H. Wolpert and W.G. Macready. No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation*, 1(1):67–82, April 1997.

[37] Ronald Van Loon. Machine learning explained: Understanding supervised, unsupervised, and reinforcement learning, 2018.

[38] Boran Sekeroglu, Shakar Sherwan Hasan, and Saman Mirza Abdullah. Comparison of Machine Learning Algorithms for Classification Problems. In Kohei Arai and Supriya Kapoor, editors, *Advances in Computer Vision*, volume 944, pages 491–499. Springer International Publishing, Cham, 2020. Series Title: Advances in Intelligent Systems and Computing.

[39] Ian J. Goodfellow, Nate Koenig, Marius Muja, Caroline Pantofaru, Alexander Sorokin, and Leila Takayama. Help me help you: Interfaces for personal robots. In *2010 5th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pages 187–188, Osaka, Japan, March 2010. IEEE.

[40] John Ashworth Nelder and Robert WM Wedderburn. Generalized linear models. *Journal of the Royal Statistical Society: Series A (General)*, 135(3):370–384, 1972. Publisher: Wiley Online Library.

[41] Leo Breiman, Jerome Friedman, Charles J. Stone, and Richard A. Olshen. *Classification and regression trees*. CRC press, 1984.

[42] Irina Rish. An empirical study of the naive Bayes classifier. In *IJCAI 2001 workshop on empirical methods in artificial intelligence*, volume 3, pages 41–46, 2001. Issue: 22.

[43] Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[44] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995. Publisher: Springer.

[45] Jerome H. Friedman. Greedy function approximation: a gradient boosting machine. *Annals of statistics*, pages 1189–1232, 2001. Publisher: JSTOR.

[46] M. Emre Celebi and Kemal Aydin. *Unsupervised learning algorithms*. Springer, 2016.

[47] Anil K. Jain and Richard C. Dubes. *Algorithms for clustering data*. Prentice-Hall, Inc., 1988.

[48] Anil K. Jain. Data clustering: 50 years beyond K-means. *Pattern Recognition Letters*, 31(8):651–666, June 2010.

[49] I K Fodor. A Survey of Dimension Reduction Techniques. Technical Report UCRL-ID-148494, 15002155, May 2002.

[50] Svante Wold, Kim Esbensen, and Paul Geladi. Principal component analysis. *Chemometrics and Intelligent Laboratory Systems*, 2(1-3):37–52, August 1987.

[51] Juan Shi, Kin Keung Lai, and Gang Chen. Dominating Factors Affecting Individual Retweeting Behavior. In *Individual Retweeting Behavior on Social Networking Sites*, pages 61–88. Springer Singapore, Singapore, 2020.

[52] Ye Pan, Feng Cong, Kailong Chen, and Yong Yu. Diffusion-aware personalized social update recommendation. In *Proceedings of the 7th ACM conference on Recommender systems - RecSys '13*, pages 69–76, Hong Kong, China, 2013. ACM Press.

[53] Preethi Lahoti, Gianmarco De Francisci Morales, and Aristides Gionis. Finding topical experts in Twitter via query-dependent personalized PageRank. In *Proceedings of the 2017 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2017*, pages 155–162, Sydney Australia, July 2017. ACM.

[54] Carmen De Maio, Giuseppe Fenza, Mariacristina Gallo, Vincenzo Loia, and Mimmo Parente. Time-aware adaptive tweets ranking through deep learning. *Future Generation Computer Systems*, 93:924–932, April 2019.

[55] Michail Vougioukas, Ion Androutsopoulos, and Georgios Paliouras. Identifying Retweetable Tweets with a Personalized Global Classifier. In *Proceedings of the 10th Hellenic Conference on Artificial Intelligence - SETN '18*, pages 1–8, Patras, Greece, 2018. ACM Press.

[56] Tim Paek, Michael Gamon, Scott Counts, David Maxwell Chickering, and Aman Dhesi. Predicting the Importance of Newsfeed Posts and Social Network Friends. In *AAAI*, volume 10, pages 1419–1424, 2010.

[57] Emilee Rader and Rebecca Gray. Understanding User Beliefs About Algorithmic Curation in the Facebook News Feed. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems - CHI '15*, pages 173–182, Seoul, Republic of Korea, 2015. ACM Press.

[58] Jill Freyne, Shlomo Berkovsky, Elizabeth M. Daly, and Werner Geyer. Social networking feeds: recommending items of interest. In *Proceedings of the fourth ACM conference on Recommender systems - RecSys '10*, page 277, Barcelona, Spain, 2010. ACM Press.

[59] Kalina Bontcheva, Genevieve Gorrell, and Bridgette Wessels. Social media and information overload: Survey results. *arXiv preprint arXiv:1306.0813*, 2013.

[60] Daniel Ramage, Susan T. Dumais, and Daniel J. Liebling. Characterizing microblogs with topic models. *ICWSM*, 10(1):16, 2010.

[61] Omar Alonso, Catherine C. Marshall, and Marc Najork. Are Some Tweets More Interesting Than Others? #HardQuestion. In *Proceedings of the Symposium on Human-Computer Interaction and Information Retrieval - HCIR '13*, pages 1–10, Vancouver BC, Canada, 2013. ACM Press.

[62] Himabindu Lakkaraju, Angshu Rai, and Srujana Merugu. Smart news feeds for social networks using scalable joint latent factor models. In *Proceedings of the 20th international conference companion on World wide web - WWW '11*, page 73, Hyderabad, India, 2011. ACM Press.

[63] Ibrahim Uysal and W. Bruce Croft. User oriented tweet ranking: a filtering approach to microblogs. In *Proceedings of the 20th ACM international conference on Information and knowledge management - CIKM '11*, page 2261, Glasgow, Scotland, UK, 2011. ACM Press.

[64] Kailong Chen, Tianqi Chen, Guoqing Zheng, Ou Jin, Enpeng Yao, and Yong Yu. Collaborative personalized tweet recommendation. In *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval - SIGIR '12*, page 661, Portland, Oregon, USA, 2012. ACM Press.

[65] Wei Feng and Jianyong Wang. Retweet or not?: personalized tweet re-ranking. In *Proceedings of the sixth ACM international conference on Web search and data mining - WSDM '13*, page 577, Rome, Italy, 2013. ACM Press.

[66] Qi Zhang, Yeyun Gong, Jindou Wu, Haoran Huang, and Xuanjing Huang. Retweet Prediction with Attention-based Deep Neural Network. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*, pages 75–84, Indianapolis Indiana USA, October 2016. ACM.

[67] Guangyuan Piao and John G. Breslin. Learning to Rank Tweets with Author-Based Long Short-Term Memory Networks. In *International Conference on Web Engineering*, pages 288–295. Springer, 2018.

[68] Zhaohui Zheng, Keke Chen, Gordon Sun, and Hongyuan Zha. A regression framework for learning ranking functions using relative relevance judgments. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval - SIGIR '07*, page 287, Amsterdam, The Netherlands, 2007. ACM Press.

[69] Rinkesh Nagmoti, Ankur Teredesai, and Martine De Cock. Ranking Approaches for Microblog Search. In *2010 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology*, pages 153–157, Toronto, AB, Canada, August 2010. IEEE.

[70] Edward A. Bender. *An introduction to mathematical modeling*. Courier Corporation, 2012.

[71] Warren S. McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. *The Bulletin of Mathematical Biophysics*, 5(4):115–133, December 1943.

[72] Shai Shalev-Shwartz and Tong Zhang. Stochastic dual coordinate ascent methods for regularized loss minimization. *Journal of Machine Learning Research*, 14(Feb):567–599, 2013.

[73] Wayne W. Eckerson. Predictive analytics. *Extending the Value of Your Data Warehousing Investment. TDWI Best Practices Report*, 1:1–36, 2007.

[74] Charles Nyce and API CPCU. Predictive analytics white paper. *American Institute for CPCU. Insurance Institute of America*, pages 9–10, 2007.

[75] A. H. Maslow. A theory of human motivation. *Psychological Review*, 50(4):370–396, 1943.

[76] Roy Ka-Wei Lee, Tuan-Anh Hoang, and Ee-Peng Lim. Discovering Hidden Topical Hubs and Authorities in Online Social Networks. In *Proceedings of the 2018 SIAM International Conference on Data Mining*, pages 378–386. SIAM, 2018.

[77] Zeynep Zengin Alp and Şule Gündüz Öğüdücü. Identifying topical influencers on twitter based on user behavior and network topology. *Knowledge-Based Systems*, 141:211–221, February 2018.

[78] Claudia Wagner, Vera Liao, Peter Pirolli, Les Nelson, and Markus Strohmaier. It's Not in Their Tweets: Modeling Topical Expertise of Twitter Users. In *2012 International Conference on Privacy, Security, Risk and Trust and 2012 International Confernece on Social Computing*, pages 91–100, Amsterdam, Netherlands, September 2012. IEEE.

[79] Q. Vera Liao, Claudia Wagner, Peter Pirolli, and Wai-Tat Fu. Understanding experts' and novices' expertise judgment of twitter users. In *Proceedings of the 2012 ACM annual conference on Human Factors in Computing Systems - CHI '12*, page 2461, Austin, Texas, USA, 2012. ACM Press.

[80] Yu Xu, Dong Zhou, and Séamus Lawless. Inferring your expertise from Twitter: combining multiple types of user activity. In *Proceedings of the International Conference on Web Intelligence*, pages 589–598, Leipzig Germany, August 2017. ACM.

[81] Yu Xu, Dong Zhou, and Seamus Lawless. Inferring Your Expertise from Twitter: Integrating Sentiment and Topic Relatedness. In *2016 IEEE/WIC/ACM International Conference on Web Intelligence (WI)*, pages 121–128, Omaha, NE, USA, October 2016. IEEE.

[82] Sami Belkacem, Omar Boussaid, and Kamel Boukhalfa. Ranking news feed updates on social media: A comparative study of supervised models. In *EGC - Conference on Knowledge Extraction and Management*, volume 36, pages 499–506. Revue des Nouvelles Technologies de l'Information, 2020.

[83] Sami Belkacem, Kamel Boukhalfa, and Omar Boussaid. Expertise-aware news feed updates recommendation: a random forest approach. *Cluster Computing*, 23(3):2375–2388, September 2020.

[84] Joachim Daiber, Max Jakob, Chris Hokamp, and Pablo N. Mendes. Improving efficiency and accuracy in multilingual entity extraction. In *Proceedings of the 9th International Conference on Semantic Systems - I-SEMANTICS '13*, page 121, Graz, Austria, 2013. ACM Press.

[85] Pavan Kapanipathi, Prateek Jain, Chitra Venkataramani, and Amit Sheth. User Interests Identification on Twitter Using a Hierarchical Knowledge Base. In David Hutchison, Takeo Kanade, Josef Kittler, Jon M. Kleinberg, Alfred Kobsa, Friedemann Mattern, John C. Mitchell, Moni Naor, Oscar Nierstrasz, C. Pandu Rangan, Bernhard Steffen, Demetri Terzopoulos, Doug Tygar, Gerhard Weikum, Valentina Presutti, Claudia d'Amato, Fabien Gandon, Mathieu d'Aquin, Steffen Staab,

and Anna Tordai, editors, *The Semantic Web: Trends and Challenges*, volume 8465, pages 99–113. Springer International Publishing, Cham, 2014.

[86] Wei Wei, Gao Cong, Chunyan Miao, Feida Zhu, and Guohui Li. Learning to Find Topic Experts in Twitter via Different Relations. *IEEE Transactions on Knowledge and Data Engineering*, 28(7):1764–1778, July 2016.

[87] Yajuan Duan, Long Jiang, Tao Qin, Ming Zhou, and Heung-Yeung Shum. An empirical study on learning to rank of tweets. In *Proceedings of the 23rd International Conference on Computational Linguistics*, pages 295–303. Association for Computational Linguistics, 2010.

[88] Lars Backstrom. Serving a Billion Personalized News Feeds. In *Proceedings of the Ninth ACM International Conference on Web Search and Data Mining - WSDM '16*, pages 469–469, San Francisco, California, USA, 2016. ACM Press.

[89] Leo Breiman and Adele Cutler. *Random forests-classification description: Random forests, Available at www.stat.berkeley.edu/~breiman/RandomForests/cc_home.htm [Accessed September 20, 2017]*. 2007.

[90] Michael Färber, Basil Ell, Carsten Menne, and Achim Rettinger. A comparative survey of dbpedia, freebase, opencyc, wikidata, and yago. *Semantic Web Journal*, 1:1–5, 2015.

[91] Claude Sammut and Geoffrey I. Webb. *Encyclopedia of machine learning*. Springer Science & Business Media, 2011.

[92] James Bergstra and Yoshua Bengio. Random search for hyper-parameter optimization. *Journal of machine learning research*, 13(Feb):281–305, 2012.

[93] Sami Belkacem, Kamel Boukhalfa, and Omar Boussaid. Tri des actualités sociales: Etat de l'art et Pistes de recherche. In *Journées francophones sur les Entrepôts de Données et l'Analyse en ligne (EDA)*, volume 13, pages 85–100. Revue des Nouvelles Technologies de l'Information, 2017.

[94] Sami Belkacem and Kamel Boukhalfa. Ranking News Feed Updates on Social Media: A Review and Expertise-Aware Approach. *International Journal of Data Warehousing and Mining (IJDWM)*, 17(1):15–38, 2021. Publisher: IGI Global.

[95] Sami Belkacem, Kamel Boukhalfa, and Omar Boussaid. Leveraging expertise in news feeds: A Twitter case study. In *Journées francophones sur les Entrepôts de Données et l'Analyse en ligne (EDA)*, volume 14, pages 1–16. Revue des Nouvelles Technologies de l'Information, 2018.

[96] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. BPR: Bayesian personalized ranking from implicit feedback. *arXiv preprint arXiv:1205.2618*, 2012.

[97] Larry Wasserman. *All of Statistics: A Concise Course in Statistical Inference*. Springer Texts in Statistics. Springer New York, New York, NY, 2004.

[98] Wei-Yin Loh. Fifty years of classification and regression trees. *International Statistical Review*, 82(3):329–348, 2014.

[99] Gérard Biau and Erwan Scornet. A random forest guided tour. *TEST*, 25(2):197–227, June 2016.

[100] S.Gopal Krishna Patro and Kishore Kumar sahu. Normalization: A Preprocessing Stage. *IARJSET*, pages 20–22, March 2015.

[101] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, and Vincent Dubourg. Scikit-learn: Machine learning in Python. *the Journal of machine Learning research*, 12:2825–2830, 2011. Publisher: JMLR. org.

[102] Dian Maharani, Hendri Murfi, and Yudi Satria. Performance of Deep Neural Network for Tabular Data A Case Study of Loss Cost Prediction in Fire Insurance. *International Journal of Machine Learning and Computing*, 9(6):734–742, 2019. Publisher: International Association of Computer Science and Information Technology.

[103] G. Sahoo and Yugal Kumar. Analysis of parametric & non parametric classifiers for classification technique using WEKA. *International Journal of Information Technology and Computer Science (IJITCS)*, 4(7):43, 2012.

[104] Sami Belkacem, Kamel Boukhalfa, and Omar Boussaid. News feeds triage on social networks: A survey. In *Proceedings of The 2nd International Conference on Computing Systems and Applications (CSA)*, pages 34–43, 2016.

[105] Aditya Pal, Amaç Herdagdelen, Sourav Chatterji, Sumit Taank, and Deepayan Chakrabarti. Discovery of Topical Authorities in Instagram. In *Proceedings of the 25th International Conference on World Wide Web - WWW '16*, pages 1203–1213, Montréal, Québec, Canada, 2016. ACM Press.

[106] Xiang Li, Shaoyin Cheng, Wenlong Chen, and Fan Jiang. Novel user influence measurement based on user interaction in microblog. In *Advances in Social Networks Analysis and Mining (ASONAM), 2013 IEEE/ACM International Conference on*, pages 615–619. IEEE, 2013.

[107] Milad Malekipirbazari and Vural Aksakalli. Risk assessment in social lending via random forests. *Expert Systems with Applications*, 42(10):4621–4631, June 2015.

[108] Rehanullah Khan, Allan Hanbury, and Julian Stoettinger. Skin detection: A random forest approach. In *2010 IEEE International Conference on Image Processing*, pages 4613–4616, Hong Kong, Hong Kong, September 2010. IEEE.